**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**
**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**
**ІМЕНІ ІГОРЯ СІКОРСЬКОГО»**

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

«На правах рукопису»
УДК 004.043

«До захисту допущено»
Завідувач кафедри
_____ І.Р. Пархомей
(підпис)
“___”_____2019 р.

# Магістерська дисертація

**на здобуття ступеня магістра**

зі спеціальності 126 «Інформаційні системи та технології»

на тему: Підвищення ефективності обробки інформаційних потоків в системі документообігу університету

Виконав: студент другого курсу, групи ІК-71мн
(шифр групи)

Гарматін В’ячеслав Дмитрович _____ _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник доцент, к.т.н., доцент Корнага Я.І. _____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ Лісовиченко О.І. _____ _____
(назва розділу) (науковий ступінь, вчене звання, , прізвище, ініціали) (підпис)

Рецензент К.Т.Н., доцент Фіногенов О.Д. _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019 року

# Пояснювальна записка
# до магістерської дисертації

на тему: ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ ОБРОБКИ ІНФОРМАЦІЙНИХ ПОТОКІВ В СИСТЕМІ ДОКУМЕНТООБІГУ УНІВЕРСИТЕТУ

Київ – 2019 року

**Зміст**

# ВСТУП

Електронний документообіг представляє собою потужний інформаційний потік, який можна автоматизувати двома способами. У першому випадку проходженням і зберіганням документів займаються спеціалізовані системи. Але в організаціях завжди є набір документів, які не “лягають” в одну з існуючих інформаційних систем (ІС). Це можуть бути внутрішні звіти, організаційно-розпорядча, проектна та інша документація. В цьому випадку, рух документів може бути автоматизовано тільки за допомогою спеціалізованих систем електронного документообігу з підтримкою технології workflow.

Термін «Управління людськими ресурсами» (УЛР) – це процес, який по суті охоплює основні процедури, завдання, дії і політики в організації, яка пов'язана з її ключовим ресурсом – її співробітниками. Сфера діяльності відділу кадрів широка і включає в себе такі функції, як підбір персоналу, виплати співробітникам, навчання і культура роботи.

На сучасних робочих місцях недостатньо просто виконувати завдання, пов'язані з людьми, такі як управління зарплатами і проведення співбесід. Вкрай важливо, щоб робота, виконана у відділі кадрів, повністю інтегрувалася в бізнес і активно сприяла досягненню його стратегічних цілей. Таким чином, процес управління людськими ресурсами відійшов від просто орієнтованої на завдання і процеси функції до стратегічної важливості.

В умовах сучасного розвитку бізнесу Система електронного документообігу (СЕД) повинна вирішувати завдання, пов'язані з оптимізацією витрат, з можливістю економії внутрішніх ресурсів підприємства. При цьому найбільш оптимальним варіантом є така ситуація для підприємства, коли впроваджена інформаційна система електронного документообігу дозволяє швидко (за 2-3 місяці) окупити витрати на впровадження. Необхідною умовою для подібного впровадження є наявність на підприємстві співробітника, який володіє знаннями в області процесного управління, навичками в побудові діаграм нотацій опису бізнес-процесів і досить добре представляє бізнес-

процеси, що протікають на підприємстві, впроваджувати СЕД. Важливим підмогою можуть бути формалізовані схеми бізнес-процесів.

Впровадження СЕД завжди має вирішувати питання оптимізації бізнес-процесів і економії трудовитрат як керівної ланки, так і рядових співробітників підприємства. При цьому максимальний ефект від впровадження досягається, коли електронний документооборот функціонує в єдиному інформаційному просторі з системою управління та обліку. Така об'єднана система дозволяє вирішити значно більшу кількість завдань.

Для традиційних комплексних рішень безпеки тільки кінцеві хости можуть перевіряти достовірність і цілісність трафіку, що призводить до кількох проблем. Однією з основних проблем є низький рівень ефективності: якщо сама мережева інфраструктура знаходиться під атакою і не може доставляти пакети. Тому існує явна потреба в нових рішеннях, в яких політика безпеки застосовується при кожному переході в міру проходження пакету через хости. Якщо мережева інфраструктура може перевіряти достовірність трафіку, в мережі можуть прийматися контрзаходи проти різних атак на протяг всього шляху використання. Це дозволило б швидко і ефективно зупинити напади і збільшити вірогідність виявлення зловмисників.

Системи управління документами зазвичай забезпечують безпеку і контроль доступу до документів в обмеженому середовищі. Однак, коли документ залишає безпечне середовище, його легко змінити. Незахищені документи не дозволяють визначити, чи є документ справжнім, хто був укладачем і стверджують або чи був він змінений з моменту його створення.

Проблема збереження електронних документів від копіювання, модифікації і підробки вимагає для свого вирішення специфічних засобів і методів захисту. Одним з поширених в світі засобів такого захисту є кваліфікований електронний підпис (КЕП), який за допомогою спеціального програмного забезпечення підтверджує достовірність документа, його реквізитів і факту підписання конкретною особою.

Залежність від одних тільки КЕП викликає занепокоєння, оскільки пара ключів може бути створена людиною, яка потім обманним шляхом представляє пару ключів як належить іншій особі або організації.

**Мета дослідження:** Підвищення ефективності управлінських рішень при роботі системи електронних документів шляхом використання кваліфікованого електронного підпису.

Для досягнення мети вирішуються наступні завдання:

1. Провести аналіз предметної області;
2. Спроектувати, за допомогою методології SADT, функціональну модель автоматизацій процесу підготовки документів;
3. Спроектувати на основі проведеного аналізу наступні моделі представлення бізнес-процесів:
   - Контекстну діаграму;
   - Функціональну декомпозицію;
   - Діаграму інформаційних потоків.
4. Використати існуючі види алгоритмів підпису, хеш-функцію Argon2 для забезпечення захисту даних університету;
5. Розробити автоматизовану інформаційної системи «Проект документу» яка матиме зручний інтерфейс та буде надавати найбільш надійний спосіб створення, погодження та підписання документа з використанням кваліфікованого електронного підпису.

Актуальність роботи – впровадження системи електронного документообігу з використанням кваліфікованого електронного підпису та з урахуванням раціонального використання людських ресурсів.

При впровадженні розробленої автоматизованій системі електронного документообігу, університет:

- отримає надійний захист при створенні, погодженні та підписанні документів з використанням кваліфікованого електронного підпису;
- отримає можливість колективної роботи над документом, що було майже неможливо при паперовому діловодстві;

- зменшить час на створення, погодження та підписання документів;
- підвищить безпеку інформації та надійність розгалуження доступу до неї;
- зменшить кошти на розмноження, доставку та збереження документів, знизиться кількість коштів на спеціалізоване обладнання;
- збільшить кількість вільного простору та ефективність роботи закладу;
- підвищить надійності та зручності збереження документів, так як вони будуть зберігатись на віддаленому сервері;
- покращить контроль за виконанням документів.

**Наукова новизна**: Полягає у модифікуванні алгоритму кваліфікованого електронного підпису Ed22519 за допомогою використання хеш-функції Argon2 та застосування модифікації в автоматизованій інформаційній системі “Проект документу”.

**Об’єкт дослідження:** Система документообігу університету

**Предмет дослідження**: Створення електронної системи документообігу та процес впровадження кваліфікованого електронного підпису в електронну систему документообігу

# РОЗДІЛ 1 АНАЛІЗ ВИКОРИСТАННЯ ЕЛЕКТРОНОГО ДОКУМЕНТООБІГ

## 1.1 Процеси документообігу

Шаблони робочих процесів складаються з двох типів завдань [6]: завдання перевірки і завдання затвердження. Завдання затвердження полегшують збір підписів на документах. При виконанні завдань затвердження документи можуть бути затверджені, не затверджується та твердження може бути відкладено на більш пізній термін.

Робочі процеси може ініціювати користувач вручну або автоматично в залежності від типу документу. Робочі процеси також можуть бути ініційовані для одного документа або декількох документів відразу. Робочі процеси з декількома документами можуть містити як основні документи, так і додаткові документи, де основними є ті, які розглядаються і / або затверджуються, а додаткові служать тільки для того, щоб допомогти в процесі розгляду і / або затвердження.

Після запуску робочого процесу запускається перша дія, задана в шаблоні робочого процесу. Всі учасники першого етапу отримають повідомлення в програмному забезпеченні для керування документами і електронною поштою. Завдання також відображаються в календарі. Шаблон робочого процесу можна налаштувати таким чином, щоб один або всі користувачі, призначені першого етапу, виконали завдання. Як тільки завдання завершено, документ переходить до наступного кроку, визначеному в шаблоні робочого процесу.

Документи можуть бути вилучені, оновлені і повернуті під час робочого процесу. Стан документа також можна оновлювати, коли він знаходяться на будь-якому етапі робочого процесу, це дозволяє не зупиняти робочий процес, не скасовувати або не затверджувати документ. Робочі процеси при необхідності також можуть бути перезапущено.

За станом документа можуть стежити користувачі, які беруть участь в робочому процесі, всі спостерігачі спостерігають за робочим процесом, щоб переконатися, що всі робочі процеси виконуються коректно. Історія робочого

процесу кожного документа також записується і може бути використана для перегляду запису затверджень.

Основні елементи робочого процесу можна побачити на діаграмі нижче.

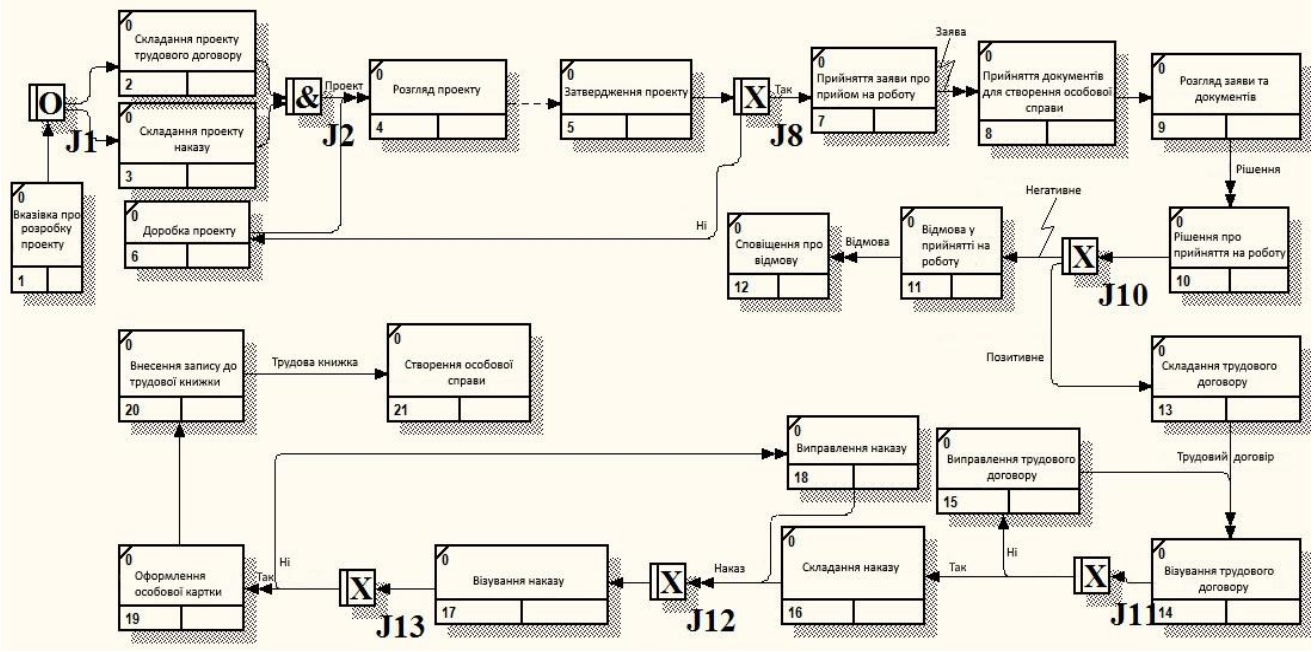


Рисунок 1.1 – Автоматизація процесу документообігу електронного документа

Процес документообігу може бути ще більш автоматизований за рахунок автоматичного запуску робочих процесів в системі управління документами. Традиційно робочі процеси в системі управління документами виконувалися вручну для кожного документа окремо. Система управління документами може автоматизувати цей ручний процес для конкретної схеми документа і шаблону робочого процесу, так що робочий процес може бути ініційований автоматично. Таким чином, всі документи певного типу, які додаються в систему, можуть автоматично «запускатися» для запуску процесу робочого процесу.

Використовуючи програмне забезпечення у відділі кадрів, яке додає автоматизацію до ключових процесів управління персоналом, університет може підвищити узгодженість і точність даних, з якими вони працюють, що призводить до зменшення кількості помилок і скорочення часу, що витрачається на трудомісткі процеси. Це може звільнити час співробітникам відділу кадрів, що дозволить їм зосередитися на інших завданнях. У систему електронного документообігу також вбудований ряд інструментів звітності,

тому університет може легко отримувати і аналізувати цінні дані, пов'язані з таким областям, як відсутність, продуктивність, плинність кадрів і т. д.

Процес управління людськими ресурсами має багато граней. Можна виділити деякі з критичних областей, за які відповідає HR-команда бізнесу:

1. Прийом на роботу;
2. Управління персоналом (відпустки, відрядження, атестації, зміни діяльності);
3. Компенсація і пільги;
4. Відповідність нормативним вимогам;
5. Управління ефективністю;
6. Підвищення кваліфікації співробітників.

Кожна заява про прийом на роботу, наказ, скарга і всі пов'язані документи повинні бути збережені в тій чи іншій формі і можуть дуже швидко заповнити багато шафи або жорсткі диски. Часто ці документи відправляються колу осіб для розгляду і схвалення і можуть бути втрачені або затримані. Рішення – система електронного управління документами, яка включає в себе організовану подачу цих документів і робочий процес, який забезпечує відстеження, може заощадити відділу кадрів значний час і гроші.

Електронний документообіг впорядковує ці незалежні процеси прозорим, динамічним і надійним чином, роблячи його ключовою частиною життєвого циклу документа. Механізм документообігу спрощує процес розгляду і затвердження документів в міру їх життєвого циклу. У спільній робочій середовищі на цій трудомісткою стадії росту документа можна домогтися найбільшої економії часу за рахунок впровадження електронного документообігу.

Впровадження електронних документів дозволяє співробітникам значно спростити і прискорити свої робочі процеси. Шаблони робочих процесів можуть бути налаштовані за допомогою встановлених правил або можуть бути зроблені неформальними, де кінцеві користувачі можуть визначати всі аспекти процесу робочого процесу.

## 1.2 Електронний цифровий підпис

Директива 1999/93 / ЕС («Директива про електронні підписи») [18] встановила правову основу для використання електронних підписів і пов'язаних з ними послуг по сертифікації для забезпечення належного функціонування внутрішнього ринку.

В міжнародній практиці існує кілька видів підписів:

- Машинописний;
- Відсканований;
- Електронне подання рукописного підпису;
- Унікальний набір символів;
- Цифрове подання характеристик, наприклад, відбитків пальців або сітківки;
- Підпис, створена криптографічними засобами.

Електронні підписи можна розділити на три групи:

- Прості електронні підписи – вони включають відскановані підписи;
- Розширені електронні підписи – вони однозначно (унікально) пов'язані з підписантами, здатні ідентифікувати підписала і пов'язані з даними в підпису, так що можна виявити будь-які зміни;
- Кваліфіковані електронні підписи – удосконалений електронний підпис, який створюється з використанням засобу кваліфікованого електронного підпису і базується на кваліфікованому сертифікаті відкритого ключа;
- Електронні підписи безпечні, якщо бізнес-процеси і технології, використовувані для їх створення також безпечні. Підписи, які використовуються для цінних даних, повинні бути надійно пов'язані з власником, щоб забезпечити необхідний рівень довіри і довіру до системи.

Електронні підписи забезпечують:

- Аутентифікацію – зв'язок підписувача з інформацією;
- Цілісність – дозволяє виявляти будь-які зміни в інформації.

Для забезпечення безпеки і юридичної сили електронної діяльності електронні підписи, безумовно, важливі, але не завжди є достатніми. Служби довіри надають:

• Електронну позначку часу – електронні дані, які пов’язують інші електронні дані з конкретним моментом часу для засвідчення наявності цих електронних даних на цей момент часу;

• Електронні печатки – електронні дані, які додаються створювачем електронної печатки до інших електронних даних або логічно з ними пов’язуються і використовуються для визначення походження та перевірки цілісності пов’язаних електронних даних;

• Електронна зареєстрована служба – це послуга, що дозволяє сторонам безпечно обмінюватися електронними даними, захищаючи дані від втрати, крадіжки, пошкодження або будь-яких несанкціонованих змін. Служба також надає докази, що відносяться до обробки переданих даних, в тому числі і аутентифікація веб-сайту – сертифікат, який дозволяє користувачам перевіряти справжність веб-сайту і його посилання на обличчя володіє веб-сайтом.

З 1 липня 2016 року в країнах Євросоюзу почав працювати регламент eIDAS (electronic IDentification, Authentication and trust Services) [16] про електронну ідентифікації та довірених послуги. Він виступив в силу після прийняття Положення (EU) № 910/2014. Регламент встановлює загальний стандарт для електронних підписів, електронних печаток, міток часу, послуг eDelivery і сертифікатів аутентифікації веб-сайтів.

На що рівнятися європейський регламент електронної ідентифікації eIDAS зображено на рис. 1.2

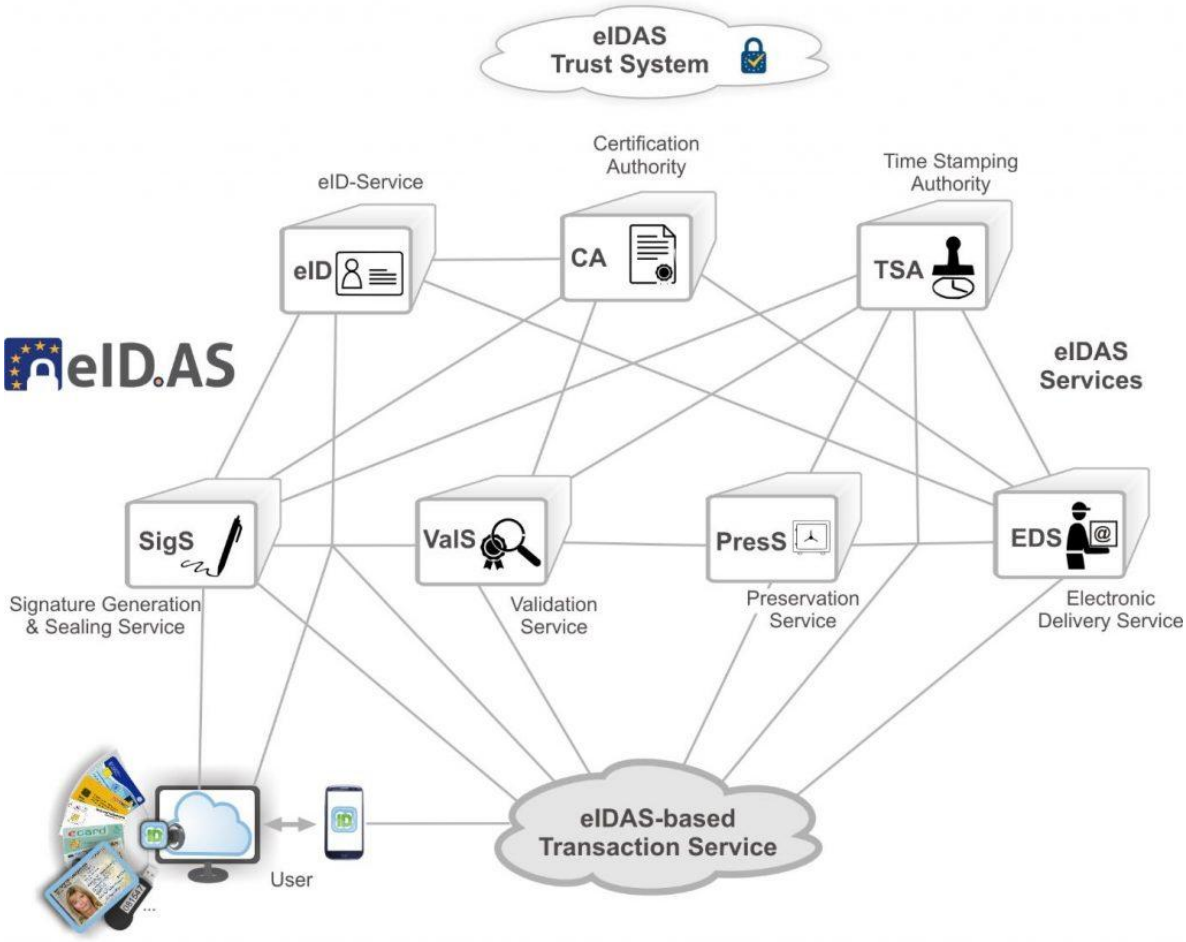


Рисунок 1.2 – eIDAS

Основні зміни в законодавстві про електронні підписи після прийняття eIDAS:

- Правовий статус закону (замість директиви) робить його безпосередньо застосовним в усій Європі без необхідності включення в національне законодавство. Таким чином, всі європейські цифрові підписи тепер узгоджені і здійснюються за єдиним стандартом;
- Можливість впровадження нових технічних рішень віддаленої підпису. Електронні документи не можуть бути позбавлені юридичної сили тільки тому, що вони в електронному вигляді;
- Впровадження електронних печаток, доступних юридичним особам, технічно схожих з електронним підписом. Вони забезпечують ідентичність і цілісність документів;
- Введення міток часу;
- Включення національних Довірених списків;
- Кваліфікований сервіс перевірки електронних підписів.

Основна мета, що стоїть за Регламентом eIDAS, полягає в оновленні цих правил і створенні єдиного режиму для взаємної електронної ідентифікації і трастових послуг на всій території ЄС.

eIDAS в основному поділений на дві частини. Перший розділ присвячений електронним системам ідентифікації та встановлює правові рамки, що дозволяють взаємно визнавати системи ідентифікації між державами-членами. Другий розділ eIDAS стосується, зокрема, трастових послуг та електронних підписів. У ній роз'яснюються існуючі правила і вводяться нові правові рамки для електронних підписів і печаток, відміток часу, зареєстрованих служб доставки і аутентифікації веб-сайтів, що забезпечує більшу юридичну визначеність для служб, які дотримуються правил eIDAS і призначені для підвищення надійності та надійності цих послуг.

1. Електронні підписи. Одним з важливих змін в eIDAS є те, що тепер електронний підпис може використовуватися тільки фізичними особами. Раніше, відповідно до Директиви про електронні підписи, електронний підпис міг використовуватися як приватними особами, так і корпоративними організаціями. Регламент eIDAS проводить відмінність між фізичними і юридичними особами

2. Вдосконалені електронні підписи. Іншою зміною від нового Регламенту є перевизначення Удосконаленої електронного підпису, яке дозволяє однозначно ідентифікувати і аутентифікувати підписувача документа і дозволяє перевіряти цілісність підписаної угоди. Ця аутентифікація зазвичай виконується шляхом видачі цифрового сертифікату центром сертифікації.

3. Кваліфікований електронний підпис. Останній тип підпису, визначений відповідно до Регламенту eIDAS, – Кваліфікований електронний підпис (QES). У той час як розширені і кваліфіковані електронні підписи однозначно пов'язані з підписує особою, кваліфіковані електронні підписи є розширені електронні підписи, створені кваліфікованими пристроями для створення електронного підпису на основі кваліфікованих сертифікатів. Кваліфіковані сертифікати можуть бути видані тільки кваліфікованим постачальником трастових послуг,

якому Наглядова орган (Supervisory Body) видав дозвіл. Дані для створення електронного підпису також повинні зберігатися на відповідному пристрої для створення підпису, такому як смарт-карта, USB-токен або хмарна служба довіри.

4. Електронні печатки. Регламент eIDAS також вводить поняття електронних печаток. Вони схожі на електронні підписи, але доступні тільки для юридичних осіб.

5. Юридична сила електронних підписів, печаток, відміток часу, зареєстрованих служб доставки і електронних документів. Статті 25, 35, 41, 43 і 46 Регламенту eIDAS [18] передбачають узгоджену і відповідну правову базу для використання електронних підписів, трастових послуг та електронних документи, забезпечивши визнання всіх в якості доказів в судочинстві.

6. Різниця між цифровим підписом і електронним підписом. Електронний підпис і цифровий підпис часто використовуються як взаємозамінні, але правда в тому, що ці два поняття різні. Основна відмінність між ними полягає в тому, що цифровий підпис в основному використовується для захисту документів і затверджується сертифікаційними органами, тоді як електронний підпис часто пов'язана з договором, в якому підписувач має намір затвердити.

7. Основні характеристики цифрового підпису. Цифровий підпис характеризується унікальною функцією в цифровій формі, такий як відбиток пальця (унікальний ідентифікатор, тд), який вбудований в документ. Підписуюча особа повинна мати цифровий сертифікат, щоб його можна було пов'язати з документом. Цифровий підпис часто стверджується сертифікаційними органами, які несуть відповідальність за надання цифрових сертифікатів, які можна порівняти з ліцензіями або паспортами. Цифровий сертифікат використовується для перевірки автентичності документа, щоб упевнитися в його автентичності, якщо він не був підроблений. Це відіграє ключову роль в перевірці особистості людини з підписом.

Іншою ключовою особливістю цифрового підпису є те, що він використовується для захисту цифрових документів. Коли документ

захищений, він може бути доступний тільки уповноваженій особі для будь-яких змін або доповнень.

Коли цифровий підпис застосовується до певного документу, цифровий сертифікат прив'язується до даних, підписуючи унікальний відбиток. Ці два компоненти цифрового підпису унікальні, і це робить її більш життєздатною, ніж мокрі підписи, оскільки його походження може бути визначене. Ця криптографічна операція допомагає виконувати наступні функції:

- Довести справжність документа і його джерела;
- Переконатися, що документ не був змінений.
- Надати ідентифікацію осіб, що підписують

## 1.3 Основні характеристики електронного підпису

Електронний підпис описується як будь-який електронний символ, процес або звук, пов'язаний із записом або контрактом, коли є намір підписати документ. Таким чином, основною особливістю електронного підпису є намір підписати документ або договір. Іншим примітним аспектом, який відрізняє електронний підпис від цифрового підпису, є те, що електронний підпис може бути усним, простим клацанням (чекбокс) або будь-яким іншим видом електронних даних.

Іншим аспектом електронного підпису є те, що він допомагає перевірити документ. Якщо він був підписаний, його справжність може бути перевірена в тих випадках, коли можуть бути виявлені беруть участь сторони. Проте, електронний документ може бути важко перевірити, враховуючи, що цифровий сертифікат, аналогічний тому, який виданий для цифрового підпису, не надається.

Іншою примітною особливістю електронного підпису є те, що він використовується для виконання угоди. Наприклад, в контракті двоє людей зазвичай погоджуються виконувати певні обов'язки, і ця угода може стати юридично обов'язковим тільки після його підписання обома сторонами. Крім

того, можна помітити, що електронні підписи зазвичай використовуються в контрактах в силу того, що вони прості у використанні.

Таблиця 1.1 – Основні відмінності електронного та цифрового підпису

| Цифровий підпис | Електронний підпис |
|---|---|
| Використовується для захисту та валідації документу | Використовується для перевірки документу |
| Підтверджується та регулюється спеціальними органами | Зазвичай не підтверджується |
| Може бути веріфікований | Підпис не може бути веріфікований |

## 1.4 Електронний цифровий підпис, як засіб захисту інформації

Системи управління документами зазвичай забезпечують безпеку і контроль доступу до документів в обмеженому середовищі. Однак, коли документ залишає безпечне середовище, його легко змінити. Незахищені документи не дозволяють визначити, чи є документ справжнім, хто був укладачем і стверджують або чи був він змінений з моменту його створення.

Проблема збереження електронних документів від копіювання, модифікації і підробки вимагає для свого вирішення специфічних засобів і методів захисту. Одним з поширених в світі засобів такого захисту є кваліфікований електронний підпис (КЕП), який за допомогою спеціального програмного забезпечення підтверджує достовірність документа, його реквізитів і факту підписання конкретною особою.

Засвідчуваний центр надає кваліфіковану електронну довірчу послугу формування, перевірки та підтвердження чинності кваліфікованого сертифіката електронного підпису чи печатки кваліфікованим надавачам електронних

довірчих послуг з використанням самопідписаного сертифіката відкритого ключа засвідчуваного центру.

Кваліфікований надавач електронних довірчих послуг – юридична особа незалежно від організаційно-правової форми та форми власності, фізична особа – підприємець, яка надає одну або більше електронних довірчих послуг, діяльність якої відповідає вимогам цього Закону та відомості про яку внесені до Довірчого списку

Засвідчуваний центр та кваліфіковані надавачі електронних довірчих послуг, стосовно яких засвідчуваним центром прийнято рішення про внесення відомостей про них до Довірчого списку, мають такі самі взаємні права та обов'язки, як і центральний засвідчуваний орган та кваліфіковані надавачі електронних довірчих послуг, стосовно яких центральним засвідчуваним органом прийнято рішення про внесення відомостей про них до Довірчого списку.

Сучасні інформаційні системи дозволяють організаціям підвищити свою ефективність, істотно скоротити їх витрати і відповідати нормативним вимогам. Система управління документами потрібує заходів для забезпечення захисту даних від НЕ авторизованого доступу та підробки.

Використання електронних підписів створює значні проблеми щодо особистості. Використання паперових коштів для створення і ведення записів часто включає ручні підписи, і такі кошти перевірки, як друк, є переважаючим засобом виконання офіційних дій. Типовими прикладами паперових правил є формальні правові вимоги на користь паперових документів і рукописних підписів або правила архівування, що вимагають зберігання цінної інформації на папері. Ці правила можуть бути знайдені в різних національних, міжнародних та наднаціональних правових рамках.

Традиційно рукописний підпис є достатнім засобом аутентифікації. Підписуючи паперовий документ, виробник «ідентифікує» себе як автора документа і підтверджує «цілісність» документа. Акт про електронні комунікації вказує на намір бути пов'язаним зі змістом документа. Процедура

підписання також тягне за собою можливість відображення і служить застереженням, а також підтверджує той факт, що інформація була надана остаточна форма. Знаки відмінності можуть бути закодовані в самій інформації, щоб ідентифікувати джерело і аутентифікувати зміст. В даний час використовуються багато форм цифрової аутентифікації, такі як використання пароля, такого як PIN-код, використання методів шифрування, таких як цифрові підписи, і використання біометричної ідентифікації, такий як відбитки пальців або розпізнавання голосу. В основному ці методи аутентифікації об'єднуються, забезпечуючи високий рівень безпеки.

Проблема ідентифікації викликає стурбованість у зв'язку з недостатнім захистом даних. При цьому Європейська директива вимагає загального відповідності Директиві про захист даних (95/46 / ЕС). Директива про електронні підписи також вимагає від держав-членів забезпечити, щоб постачальники послуг, що видають сертифікати для громадськості, не збирали особисті дані, крім як безпосередньо від суб'єкта даних, або без явного згоди суб'єкта даних. Існує ще одна вимога, що дані можуть збиратися тільки остільки, оскільки це необхідно для цілей видачі та обслуговування сертифіката.

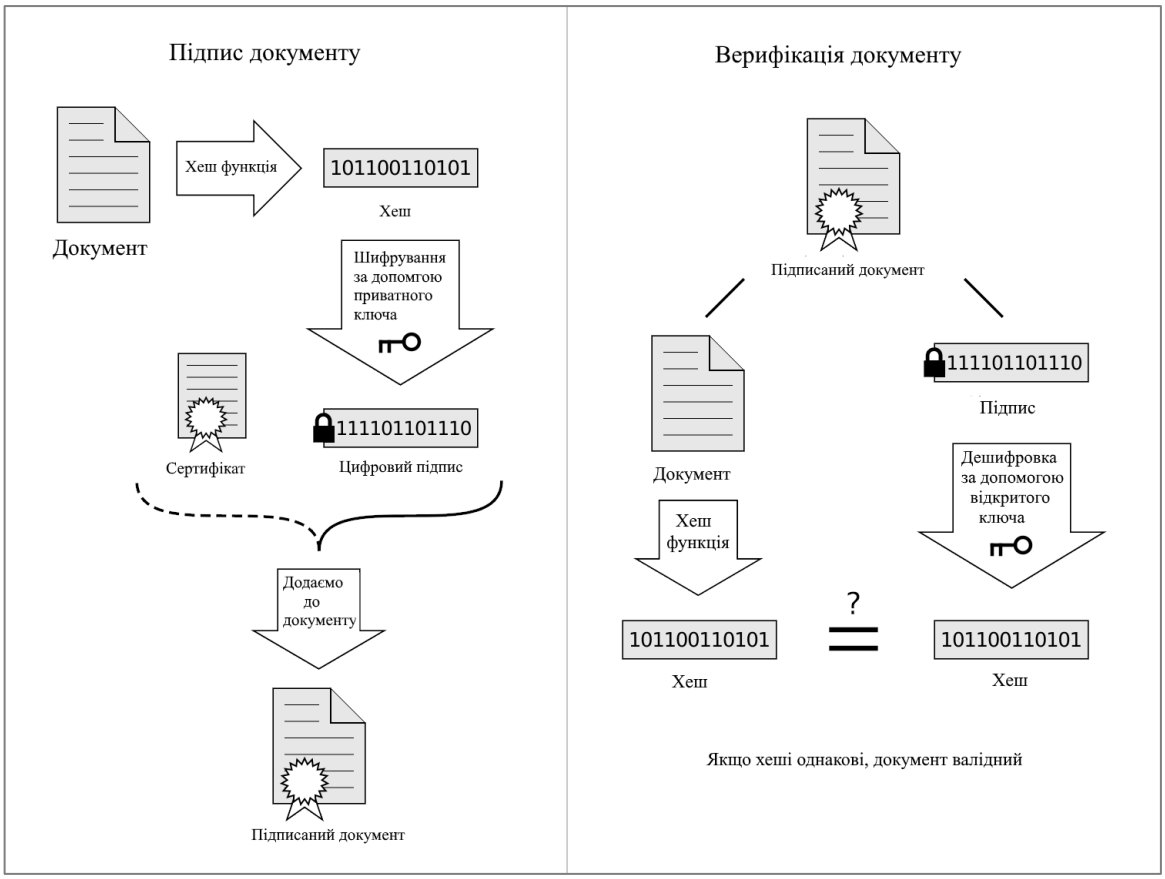


Рисунок 1.3 – Підпис та верифікація документу

Існує кілька схем побудови цифрового підпису [22]:

• На основі алгоритмів симетричного шифрування. Дана схема передбачає наявність у системі третьої особи, що користується довірою обох сторін. Авторизацією документа є сам факт зашифровування його секретним ключем і передача його довіреній особі.

• На основі алгоритмів асиметричного шифрування. На даний момент такі схеми найбільш поширені і знаходять широке застосування.

• Комбіновані.

Симетричне шифрування. Шифрування, в якому для шифрування і розшифрування застосовується один і той же криптографічний ключ. Величезним плюсом такого підходу є швидкість шифрування, але обидві сторони, між якими інформація пересилається, повинні знати ключ. Даний вид шифрування ділиться на два види: потокові шифри, де кожен символ повідомлення шифрується по одному з відповідною цифрою потоку ключів (RC4, SCALA20, Achterbahn); блокові шифри, який розбиває повідомлення на

блоки фіксованої довжини і оперує з отриманими блоками (DES, AES, Blowfish).

Асиметричне шифрування. Шифрування, в якому існує два види ключів, відкритий і закритий ключ. Відкритий ключ передається по відкритому каналу і використовується для перевірки ЕЦП і для шифрування повідомлення. Для генерації ЕЦП і для розшифровки повідомлення використовується закритий ключ. Види асиметричних шифрів: RSA, DSA, ElGamal, Rabin.

Комбіноване шифрування. За допомогою симетричного шифрування швидко зашифровуються потрібні дані, ключ прикладається в повідомлення або ЕЦП, а ключ від симетричного шифру зашифрована асиметричним шифром.

Для формування ЕЦП потрібна хеш-функція, яка скоротить повідомлення будь-якого обсягу до певної кількості байтів. Отриманий хеш зашифровують і надають з вихідним документом на перевірку. За отриманою композиції можна довести, що документ з моменту обчислення підпису не був змінений.

Методи шифрування будуються на факторизації великих чисел (RSA, DSA, ElGamal) і дискретно логарифмування (ECDSA, EdDSA, GOST R 34.10-2012).

## 1.5 Алгоритм цифрового підпису

Ed25519 один з видів алгоритмів підпису на основі еліптичної кривої Curve25519, що належить сімейству EdDSA [13]. Крива виглядає наступним чином:

$$y^2 = x^3 + 48666 * x^2 + x \tag{1.1}$$

Це – крива Монтгомері за модулем простого числа 2255 – 19 (що і дало назву схемі) і з базовою точкою x = 9. Схема використовує точки в стислій формі (тільки X координати), дозволяючи таким чином використовувати "Сходи Монтгомері", яка робить множення точок за фіксований час, позбавляючи нас від атак за часом.

Ed25519 складається з трьох модулів:

- алгоритм цифрового підпису;

- хеш-функція SHA512;
- генератор випадкових чисел, для генерації пар ключів.

EdDSA і ECDSA вимагають генерації випадкового значення (скалярної пари ефемерних ключів) під час процесу генерації підпису [11] та секретність цього випадкового значення критична для безпеки: знання одного такого випадкового значення або часткового знання кількох з них, дозволяє відновити закритий ключ підписувача. У ECDSA не вказано, як генерувати це випадкове значення і, отже, реалізації критично покладаються на якість випадкового числа. EdDSA прибирає цю залежність шляхом детермінованого вилучення секрету з повідомлення і довгострокового допоміжного ключа з використанням криптографічного хеш-функції SHA-512.

EdDSA вважається більш стійкою до атак по стороннім каналах. Автори покладаються на ідею «генерувати випадкові підписи таємно детерміністичним способом», так що «різні повідомлення призводять до різних, важко прогнозованих значень ефемерного ключа. Кілька біт зазвичай можна отримати з атак по побічним каналам або з нерівномірності розподілу, з якого береться r, тому автори EdDSA справедливо вказують на той факт, що «детерміністична особливість» не приводить до очевидних витоків при атаках на сторонні канали.

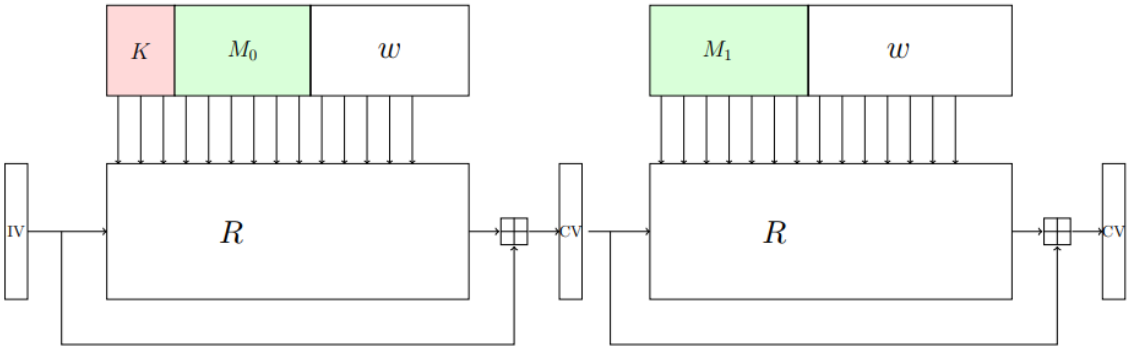


Рисунок 1.4 – Алгоритм роботи SHA-512

SHA-512 належить сімейству SHA-2, розробленим АНБ. SHA-2 – це структура Меркла-Дамгарда. Структура передбачає ітеративе поновлення значення CV, це значення ініціалізюється фіксованим початковим значенням IV. Повідомлення доповнюється (якщо потрібно) і розділяється на блоки. На кожній ітерації обробляється блок повідомлення. Малюнок показує генерацію

ефемерного скаляра, де допоміжний ключ і повідомлення хешіруются. Буквою K позначений допоміжний ключ b, $M_i$ – вхідний повідомлення, W чергу залишилися повідомлень і R функція стиснення. $M_0$ – фрагмент повідомлення, який знаходиться в тому ж блоці, що і ключ. На жаль, дана структура також вразлива для атак на сторонні канали [12].

Достатньо буде лише кілька сотень ЕЦП [12] щоб дізнатися допоміжний ключ, а за допомогою нього вже можна буде витягти закритий ключ. Хоч від таких атак і можна захиститися, налаштувавши генератор випадкових чисел так, щоб була можливість використовувати їх для підпису декількох повідомлень, це суперечить вимогам безпеки EdDSA [11].

Якщо використовувати ЕЦП для забезпечення захисту даних компанії, можна модифікувати Ed25519 за допомогою іншої хеш-функції, яка не буде піддана тих же проблем що і SHA-512.

Однією з найбільш надійних і гнучких функцій на сьогоднішній день є Argon2 [10]. Вона була розроблена для ефективного і хешування паролів. Вона націлена на високу швидкість заповнення пам'яті і ефективне використання декількох обчислювальних блоків, і в той же час забезпечує захист від безлічі видів атак. Argon2 має три варіанти: Argon2i, Argon2d і Argon2id [9]. Argon2d працює швидше і використовує доступ до пам'яті, що залежить від даних, що робить його дуже стійким до атак за допомогою GPU (brute force) і підходить для додатків, в яких відсутні загрози зі сторонніх атак. Argon2i використовує незалежний від даних доступ до пам'яті, який кращий для хешування, але він повільніше, оскільки робить більше проходів по пам'яті для захисту від компромісних атак. Argon2id – це гібрид Argon2i і Argon2d, що використовує комбінацію залежного від даних і незалежного від даних доступу до пам'яті, що дає деяку стійкість Argon2i до атак на сторонні канали і більшу частину стійкості Argon2d до атак на злом GPU.

Незалежна від даних версія Argon2i надійно заповнює пам'ять, витрачаючи 2 циклу CPU на байт, а Argon2d в три рази швидше. Це робить її

придатною для додатків, які потребують memory-hardness, але не можуть собі дозволити багато процесорного часу.

Незважаючи на високу продуктивність, Argon2 забезпечує розумний рівень стійкості [11]. З кількістю проходів за замовчуванням по пам'яті (1 для Argon2d, 3 для Argon2i) зловмисник, оснащений ASIC, не може зменшити час виконання (time-area product), якщо пам'ять менш необхідної в 4 і більше разів. Чим більше проходів по пам'яті, тим серйозніші штрафи накладатимуться.

Таблиця 1.2 – Штрафи на читання при зменшенні необхідної кількості пам'яті

| Частина необхідної пам'яті | $\frac{1}{2}$ | $\frac{1}{3}$ | $\frac{1}{4}$ | $\frac{1}{5}$ | $\frac{1}{6}$ |
|---|---|---|---|---|---|
| 1 прохід | 1.7 | 3 | 6.3 | 16.6 | 55 |
| 2 проходи | 15 | 410 | 19300 | $2^{20}$ | $2^{25}$ |
| 3 проходи | 3423 | $2^{20}$ | $2^{20}$ | – | – |

Таблиця 1.3 – Штрафи на час при зменшенні необхідної кількості пам'яті

| Частина необхідної пам'яті | $\frac{1}{2}$ | $\frac{1}{3}$ | $\frac{1}{4}$ | $\frac{1}{5}$ | $\frac{1}{6}$ |
|---|---|---|---|---|---|
| 1 прохід | 2.7 | 3.5 | 4.8 | 6.7 | 9.2 |
| 2 проходи | 6.7 | 13.3 | 27.8 | 48 | 74 |
| 3 проходи | 21.7 | 57 | 104 | – | – |

Argon2 масштабується як у часі, так і в обсязі пам'яті. Обидва параметра можуть бути змінені незалежно, за умови, що для заповнення пам'яті завжди потрібна певна кількість часу.

Argon2 може використовувати до $2^{24}$ потоків паралельно, хоча 8 потоків вичерпують доступну пропускну здатність і обчислювальну потужність середньостатистичного ПК [7].

Таблиця 1.4 – Характеристики середньостатистичного ПК

| | |
|---|---|
| Версія ОС | Windows 10 x64 |
| RAM | 8 GB |
| Кількість фізичних ядер | 4 |
| Тактова частота | 3.3 Ghz |

Argon2 оптимізований для архітектури x86, тому його впровадження на спеціальному обладнанні для злому не буде ні дешевим, ні швидким. Навіть спеціалізовані ASIC будуть вимагають значних площ і не дозволять скоротити час виконання (time-area product).

Argon2 підтримує додаткові вхідні дані, що відокремлені від хешуємого повідомлення і nonce, такі як секретний ключ, параметри середовища, призначені для користувача дані і т. д.

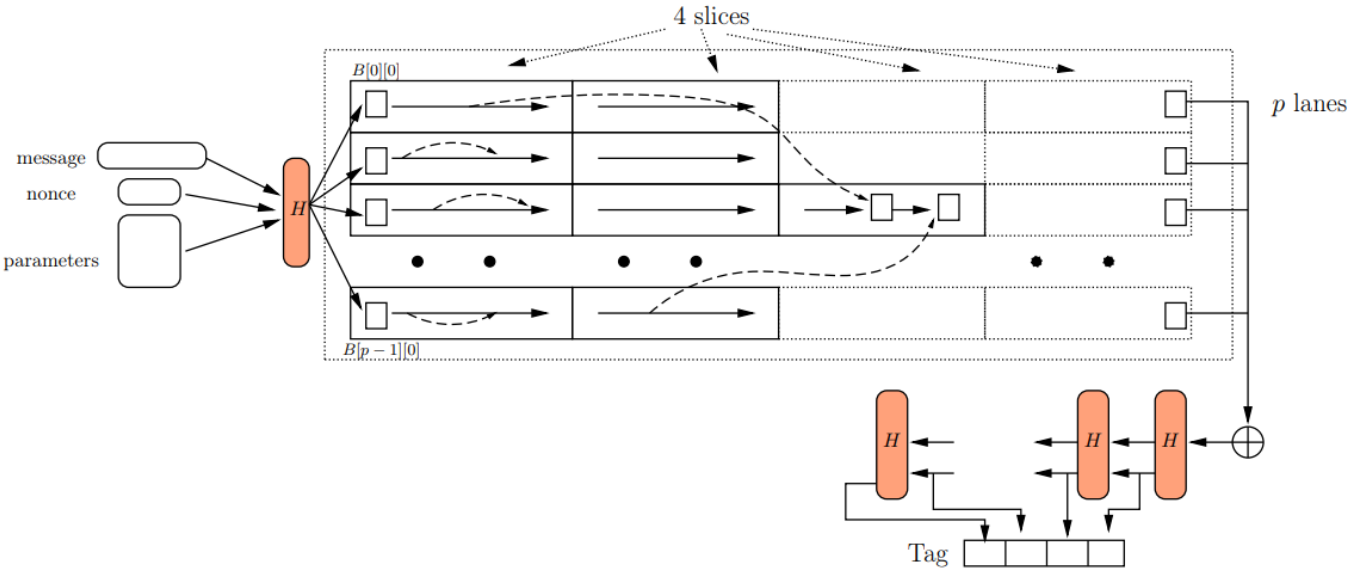


Рисунок 1.5 – Алгоритм роботи Argon2

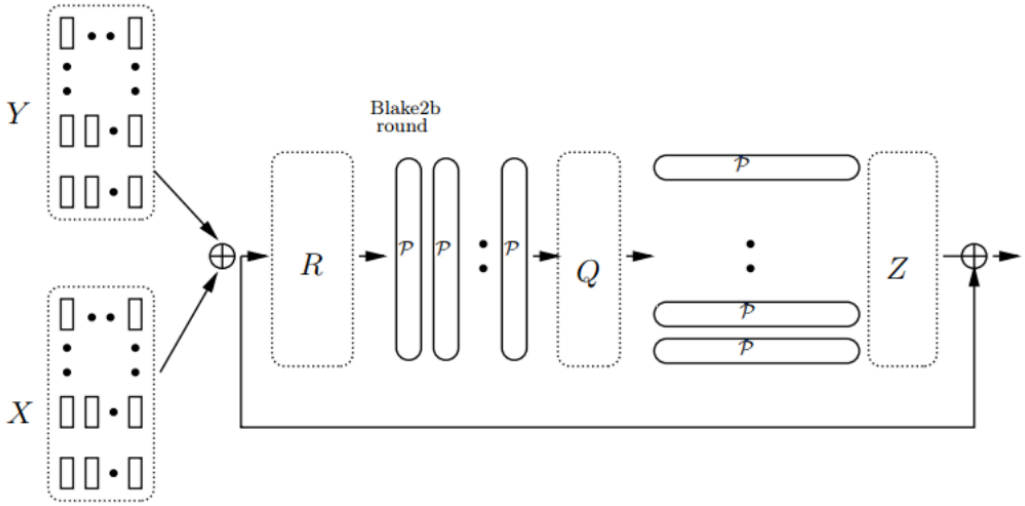


Рисунок 1.6 – Алгоритм функції і стиснення

Спершу Argon2 хешує пароль з використанням хеш-функції Blake2b [9]. Результат хешування записується в блоки пам'яті, які перетворюються з використанням функції стиснення G (вона приймає на вхід два 8192-бітних блоку, а видає 1024-бітний блок), і в результаті генерується ключ.

Завдяки Argon2, можна убезпечити закритий ключ, не підвищуючи витрат обчислювальної потужності, а завдяки Curve25519 розмір підпису буде всього лише 512 біт.

## Висновки до розділу

В розділі було проведено аналіз:

- елементів робочого процесу системи електронного документообігу;
- європейської практики використання електронних підписів і пов'язаних з ними послуг по сертифікації;
- основних особливостей електронного підпису;
- використання електронного цифрового підпису, як засобу захисту інформації;
- методів шифрування.

За результатами проведеного аналізу було зроблено висновок, що найбільш надійних і гнучких функцій для досягнення поставленої мети є Argon2.

# РОЗДІЛ 2 СИСТЕМИ УПРАВЛІННЯ ДОКУМЕНТАМИ

## 2.1 Оптимізація управління життєвим циклом інформації

Якщо ви створите надійне сховище, розмістить його в дата-центрі, заплатіть за електрику і кондиціонування, це буде обходитися вам не так вже дешево. Так, витрати на зберігання будуть з кожним роком знижуватися на 20% [1], але і кількість вашої інформації також буде з кожним роком зростатиме на 40% [1]. Витрачати на зберігання ви будете не менше, ніж раніше – проста арифметика!

Інформаційне ожиріння має ряд побічних ефектів. Навіть якщо вам вдалося владнати фінансове питання, ваша організація все одно буде схильна до серйозного ризику.

По-перше, зайва інформація являє собою серйозну загрозу для бізнесу. Через велику достатку невикористаної і непотрібної інформації рух інформаційних потоків всередині організації сповільнюється. Спираючись на застарілу, що втратила актуальність інформацію, можна прийняти неправильне рішення. Накопичення інформації не дозволить оптимізувати бізнес-процеси, поліпшити маркетинг, підвищити якість послуг, налагодити комунікації з клієнтами. Крім того, на зберігання зайвої інформації йдуть суми, яким можна було б знайти більш корисне застосування.

По-друге, надлишок інформації породжує ряд проблем юридичного характеру. Всі інформація організації в електронному вигляді може бути використана в судових розглядах. Юридичний відділ організації повинен вести облік всієї юридично значимої інформації. Якщо інформації занадто багато, то єдиним рішенням у такій ситуації буде збереження всієї інформації, контроль доступу до неї і прийняття певних заходів щодо захисту. Це неефективно, до того ж забирає багато часу і коштів. Якщо раптом буде потрібно пред'явити необхідну доказову інформацію в суд, доведеться витратити дуже багато часу, перш ніж ви знайдете те, що дійсно потрібно.

По-третє, надмірна інформація породжує проблеми з забезпеченням відповідності нормам і стандартам (compliance). Різні нормативні документи передбачають, що всі матеріали, що мають статус записів, повинні зберігається протягом строго обмеженого періоду часу. Існують також вимоги щодо того, як слід розпоряджатися документами після закінчення строків зберігання. Фахівцям з управління записами в своїй роботі доводиться враховувати безліч вимог, що виходять від різних інстанцій і стосуються часом сотень тисяч і мільйонів записів. Якщо в організації занадто багато несистематизованого контенту, то його навряд чи можна буде класифікувати, розробити графіки зберігання і т. п. Проведення аудитів на відповідність нормам стає вкрай скрутній завданням, з'їдає дуже багато часу. Крім того, невідповідність внутрішнім і галузевими нормами для організації ще й загрожує фінансовими і репутаційні втратами.

Отже, зменшити кількість споживаної інформації не вийде. Як же в такому випадку боротися з інформаційним ожирінням? Тут можна виділити два підходи. Перший з являє собою симптоматичну терапію:

• Дедуплікація, звичайно, допоможе виявити численні копії одного і того ж документа. Вона ефективна, якщо ви можете застосувати її по відношенню до всіх сховищ документів (ECM-система, управління записами, жорсткі диски персональних комп'ютерів, поштові сервери і т. П.). На практиці ситуація виглядає інакше: дедуплікація зазвичай зачіпає два-три джерела і застосовується до файлів, але не до структурованих даних;

• Архівація і структуроване зберігання. Грамотний підхід до вибору способу зберігання архівуються даних дозволяє істотно знизити витрати. Зовсім не обов'язково зберігати все на дорогому обладнанні. Більшу частину даних організації цілком можна зберігати на недорогому обладнанні. І не так уже й важливо, що відновлення цих даних відбудеться не миттєво, а займе кілька годин або навіть днів. Але як визначити, де саме потрібно зберігати ту чи іншу інформацію? Більшість організацій використовують дорогі і надійні сховища для найважливіших інформаційних систем незалежно від значущості та

актуальності збережених в них даних. При цьому до мережевих дисків, на яких зберігається величезна кількість файлів організації, навряд чи застосовна ідея структурованого зберігання;

• Стиснення. У всіх системах зберігання зазвичай використовуються складні алгоритми для скорочення дискового простору, що стискають збережені дані і розтискати їх при добуванні. Вони зазвичай дуже дорогі і вимогливі до апаратних ресурсів: інакше неможливо забезпечити прийнятну швидкість роботи.

Всі ці процедури, звичайно, дають певний ефект, але вельми незначний. Вони, звичайно, можуть призвести до певного зниження витрат, але не можуть знизити темпи зростання інформації. За допомогою цих процедур можна вирішити юридичні проблеми, що виникають внаслідок інформаційного ожиріння. Керувати вручну зберіганням інформації та одночасно стежити за забезпеченням відповідності всім законодавчим, галузевим і внутрішнім нормам вкрай складно.

Потрібна така політика управління життєвим циклом інформації, яка дозволила б відслідковувати дотримання всіх графіків зберігання, регулювати інформаційний метаболізм і забезпечувати своєчасну утилізацію інформаційного сміття. У більшості організацій така робота взагалі не ведеться.

Все, здавалося б просто. Але як же виробити політику, яка дозволила б виявити ті 30% інформації [1], які потрібно зберігати?

Дослідження показують, що можна виділити три причини, за якими організаціям потрібно зберігати інформацію протягом певного періоду часу:

• нормативні зобов'язання (їх виконання контролюється фахівцями з управління записів);

• юридичні зобов'язання (їх виконання контролюється юридичним відділом);

• бізнес-значущість (перевірка на бізнес-значущість проводиться кожним підрозділом організації).

Таким чином, три зазначених вище пункту відповідають за швидкість інформаційного метаболізму. Але в той же час всі ці підрозділи практично не контактують один з одним, говорять на різних мовах, у них немає спільних політик і процедур контролю, на основі яких вони можуть вибудовувати комунікацію з IT.

Юристи оцінюють інформацію на предмет юридичної значимості. Фахівці з управління записами розробляють таксономії, визначають терміни і графіки зберігання. А у інших підрозділів є справи важливіші, і у них немає особливого бажання стежити за дотриманням термінів або відповідністю законодавчим нормам. Але їм дуже потрібно, щоб важлива інформація завжди була доступна, щоб на її основі можна було приймати рішення і щоб вона була підмогою в роботі з клієнтами.

В основі управління життєвим циклом інформації повинна лежати уніфікована політика. Є загальний репозиторій, в якому фахівці з управління записами зберігають і видаляють документи відповідно до прийнятих графіками. Юристи коригують графіки зберігання виходячи з міркувань юридичної значимості документів. Представники інших підрозділів користуються зберігається в репозиторії інформацією і створюють нову. І все при цьому взаємодіють з IT-відділом.

Як тільки розроблені загальні принципи, що регулюють інформаційний метаболізм, необхідно застосовувати їх до всіх механізмів, що здійснюють фізичне управління інформацією.

Застосування єдиної політики по відношенню до всіх репозиторіїв, системам управління записами і т. п. Дозволяє відстежувати зростання даних і своєчасно здійснювати очищення організаційних ресурсів від непотрібної інформації.

## 2.2 Аналіз предметної області

Для вирішення поставлених задач був проведений аналіз процесів документообігу університету, де була виявлена можливість раціоналізації його за рахунок зменшення дублюючої інформації.

В результаті проведеного аналізу предметної була спроектована контекстна діаграма «Оформлення працівника на роботу» (рис. 2.1) [23].

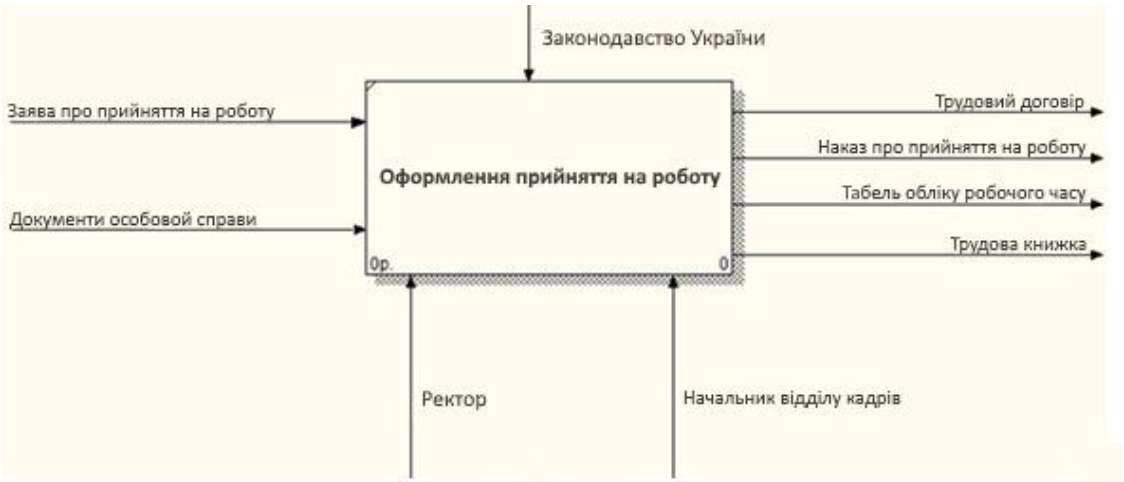


Рисунок 2.1 – Контекстна діаграма «Оформлення на роботу»

Стратегії та процедури, якими керується процес (управління) – Конституція України [17], КЗпП України [17], а також інші чинні законодавчі акти України.

Під час аналізу, була створена декомпозиція системи паперового діловодства (рис. 2.2). Вхідною інформацією для модуля «Наказ щодо прийняття» є відділи, посади, ставки, оклади і т. д.. Введення вхідної інформації здійснюється працівником відділу кадрів. Вихідною інформацією для системи є вихідні документи.

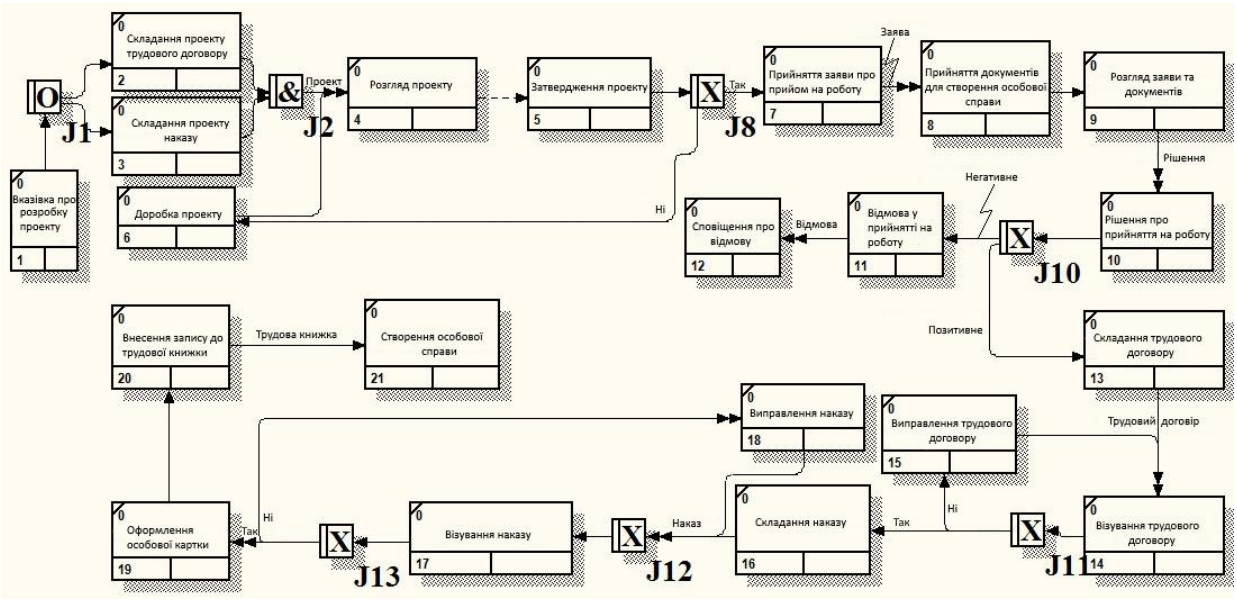


Рисунок 2.2 – Функціональна декомпозиція системи

Впроваджувати автоматизацію необхідно на всіх етапах діловодства, починаючи з підготовки документів. Діаграма інформаційних потоків модулю «Наказ на зарахування» за методологією DFD, наведено на наступному малюнку (рис. 2.3).

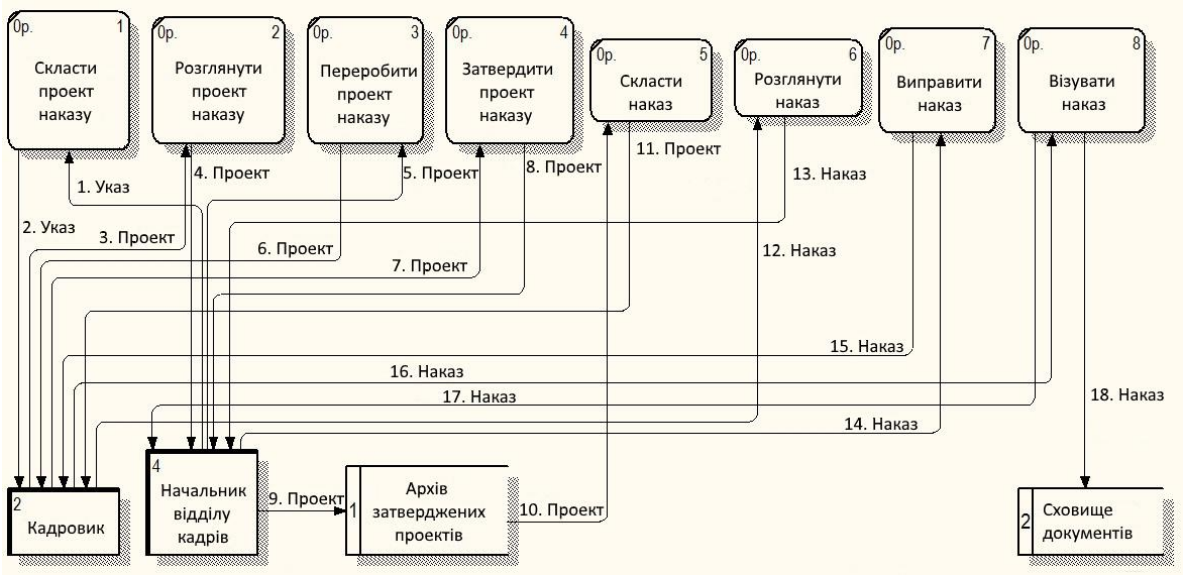


Рисунок 2.3 – Діаграма інформаційних потоків

Актуальність проблеми впровадження системи електронного документообігу визначається необхідністю створення єдиного документаційного простору з урахуванням раціонального використання людських ресурсів при виконанні діловодних робіт. Деякі дослідження проблем в області паперового діловодства виявили такі найбільші недоліки:

- Можлива втрата документів;
- Великі затрати часу на підготовку та узгодження документів;
- Велика надмірність документообігу;
- Відсутність зв’язків між документами в разі внесення зміни;
- Складність контролю;
- Фінансові витрати на закупівлю витратних матеріалів;
- Накопичення архівних матеріалів;
- Відсутність розмежування доступу до інформації і, як наслідок, її можливе розголошення.

Завдяки автоматизованій системі електронного документообігу, університет матиме багато позитивних наслідків:

- Навчальний заклад витрачатиме набагато менше коштів на копіювання, доставку та збереження документів, знизиться кількість коштів на спеціалізоване обладнання;
- Збільшиться кількість вільного простору та ефективність роботи закладу;
- З’явиться можливість колективної роботи над документом, що було майже неможливо при паперовому діловодстві;
- Значно збільшиться швидкість пошуку документів за різними параметрами;
- Збільшення безпеки інформації та надійне розгалуження доступу до неї;
- Збільшення надійності та зручності збереження документів, так як вони будуть зберігатись на віддаленому сервері;
- Покращення контролю за виконанням документів.

## 2.3 Вимоги до системи

Розподілена архітектура системи управління документами, повинна відповідати наступним вимогам:

- Масштабованість, надійність і керованість для економічного корпоративного розгортання. Це означає, що можна починати впровадження

системи покроково, а потім нарощувати до рівня підприємства й за його межі як корпоративну технологію управління документами;

• Автоматична підтримка розподіленого управління різними інформаційними матеріалами протягом їхнього життєвого циклу – від авторської розробки, створення, рецензування, узгодження, затвердження в інтерактивному режимі до розповсюдження й архівування;

• Гнучкість управління доступом до всього спектра документів: текстових, формалізованих документів, образів документів, електронних таблиць, електронної пошти;

• Зручність та логічність інтерфейсу. Проте він має бути якомога простим;

• Мінімальний вплив стороннього ПО для зручності роботи. Всі необхідні елементи мають бути включені в систему;

• Система повинна вміти синхронізувати клієнти для забезпечення актуальності та цілісності даних;

• Інтеграція з існуючими системами, що вже працюють в університеті;

• Автоматична перевірка індивідуального податкового номера на правильність вводу, перевірка його дійсності та відповідності даній особі;

• Контроль підтверджуючих документів, що надають змогу отримувати надбавки та доплати;

• Система повинна вміти знаходити людину та її особову справу, навіть якщо вона змінила прізвище, ім’я або по батькові;

• Контролювати дозвіл на роботу, якщо це необхідно. Наприклад, якщо особа молодше 14 років, то вона не має права працювати взагалі. З 14 до 16 років потрібен дозвіл від одного з батьків або одного з опікунів. Також дозвіл на роботу потрібен особам без громадянства України та іноземцям;

• Перевірка долі ставки працівника. Згідно чинного законодавства України, він не може працювати більше ніж у сумі 1.5 ставки та 240 годин.

## Висновки до розділу

За результатами проведеного аналізу процесів документообігу університету спроектовано:

- контекстну діаграму «Оформлення працівника на роботу»;
- функціональну декомпозицію системи;
- діаграму інформаційних потоків.

В результаті розроблено вимоги до системи документообігу університету.

# РОЗДІЛ 3 ПРОЕКТУВАННЯ ДОДАТКУ СИСТЕМИ ПІДГОТОВКИ ДОКУМЕНТІВ

## 3.1 Обґрунтування та вибір архітектури додатку

Архітектура додатку – система рівнів, що забезпечує взаємодію функцій програми. Найпопулярнішими архітектурами додатків є веб-орієнтована, трьох-рівнева, сервіс-орієнтована.

Веб-орієнтована архітектура, це архітектура, що призначена для створення додатку, який буде працювати у браузері або спеціально створеному додатку. Клієнт (браузер або додаток) створює HTTP запит до серверу, який повертає HTML сторінку, яку він може відобразити. На цій сторінці можуть бути розташовані елементи що взаємодіють із сервером без створення нової сторінки. Переваги: клієнту не треба оновлювати додаток, не потребує встановлення додаткового ПЗ. Недоліки: браузер не може гарантувати безпеку використання додатком (треба використовувати захищений протокол, стороннє ПЗ може змінювати вигляд HTML сторінки для використання у своїх цілях, можливий фішинг у різних його проявах), неможливість використовувати вбудовані компоненти системи, неможлива взаємодія із “залізом”.

Трьох-рівнева архітектура це клієнт-серверна архітектура системи, що складається з 3 шарів: клієнт (додаток), сервер додатку, сервер баз даних. Клієнт – компонент, який представлений на першому рівні, додаток для кінцевого користувача. На цьому рівні відсутній прямий доступ до бази даних, та бізнес логіка. Тут повинні заходитись інтерфейс користувача та модуль обміну даними із сервером. У додатку може міститися  логіка додатку (не бізнес-логіка): логіка інтерфейсної частини, взаємодія додатку з тимчасовим сховищем (локальна БД, JSON файл, тощо), перевірка введених значень, збереження та завантаження налаштувань додатку. Сервер додатку – компонент на другому рівні. На цьому рівні розміщена вся основна логіка додатку (бізнес-логіка, авторизація, тощо), модуль взаємодії з сервером баз даних та модуль взаємодії із клієнтами. Якщо серверів декілька, то додатково повинен бути

присутній модуль розподілення навантаження або окремий сервер з таким функціоналом. Сервер баз даних – компонент на третьому рівні. Тут повинні зберігатись усі дані. Дані можуть бути зовсім різними: дані для авторизації (логіни, хеші та солт паролей, різні ідентифікатори), дані потрібні для бізнес-логіки, медіа-контент. Також на цьому рівні можуть бути присутні тригери, процедури, схеми, що описують з як саме зберігаються дані у графічному вигляді. Переваги: можна забезпечити безпеку в роботі (захищений протокол передачі, перевірка самого додатку на зміну, стороннє ПЗ не може змінювати вигляд інтерфейсу, неможливий фішинг). Недоліки: потрібне встановлення на комп'ютер, може потребувати додаткового ПЗ, якщо не оновлене – може некоректно працювати.

Сервіс-орієнтована архітектура дуже схожа на трьох-рівневу архітектуру, проте є її стандартизованим аналогом. Замість серверу додатку використовується сервіс. Передача даних між сервісом та клієнтом йде на стандартній мові – XML або JSON. Це дає велику перевагу такій архітектурі через те, що не важливо на якій мові написаний клієнт, він повинен лише підтримувати передачу даних у стандартному формат. Дану архітектуру можна розділити на два підвиди за форматом передачі даних: SOAP та REST [19]. SOAP – базується на контракті, опису функцій та типів даних якими оперує сервіс та клієнт під час обміну даними з клієнтом. Якщо контракти сервісу та клієнта однакові, то вони можуть взаємодіяти. Найчастіше, цей формат використовує XML. REST, в свою чергу, базується на стандартних HTTP запитах та ресурсах. Ресурси в даному розумінні – кінцевий URL, на якому знаходиться API метод, що оброблюватиме запит. Даний формат, начастіше, використовує JSON. Сервіс-орієнтована унаслідувала переваги та недоліки трьох-рівневої архітектури, проте має і свої. Переваги: клієнт не залежить від мови програмування. Недоліки: потрібна підтримка XML/JSON, клієнт повинен представити сервісу контракт для SOAP.

Серед наведених вище архітектур, враховуючи їх переваги та недоліки, вимоги до розроблюваної системи та враховуючи аспекти бізнес-логіки, що

пред’являються до системи, тому було обрано сервіс-орієнтовану архітектуру. Завдяки сервіс-орієнтованій архітектурі вся система буде гарно масштабованою, є можливість зручно проводити тести як окремих компонентів окремого шару так і шару в цілому. База даних буде захищена від несанкціонованого доступу, та може використовувати лише обрані функції. Також перевагою буде те, що не потрібно буде оновлювати додаток при зміні бізнес-логіки. Додатковим плюсом всієї системи є те, що клієнт може бути написаній на любій мові програмування тому не залежить від операційної системи користувача. Проте треба буде розробити систему оновлення, бо через деякі зміни сервісу або системи (зміна алгоритму авторизації або зміна контракту) клієнтський додаток може перестати працювати.

## 3.2 Обґрунтування та вибір технологій програмування

### 3.2.1 Проектування бази даних

Ціллю проектування бази даних є створення детальної моделі збереження даних. Розробка БД виконується на основі предметної області. Опис предметної області, в даному випадку, повинен охоплювати всі сутності, що мають зберігатись в базі даних та забезпечувати логічний зв’язок між цими даними. Тобто, в результаті розробки БД ми повинні визначити дані, що будуть там зберігатись, зв’язки між цими даними та накласти логічну структуру на дані.

Розробку архітектури варто почати з концептуального проектування. Концептуальне проектування це побудова моделі даних якнайбільш абстрактно. На цьому етапі ми повинні визначити які саме об’єкти будуть зберігатись в БД, які зв’язки між ними та вимоги до допустимого діапазону значень даних.

Наступним етапом буде логічне проектування. На цьому етапі ми повинні створити опис схеми бази даних на основі обраної моделі даних. Наприклад, для реляційної моделі опис буде складатись із таблиць та відношень. Відношення, в свою чергу, складаються із первинних, унікальних та вторинних ключів.

Останнім етапом буде створення цього опису під конкретну СУБД. Цей етап називається фізичне проектування. Результатом буде код, який написаний на мові DDL з урахуванням особливостей обраної СУБД.

Також, при проектуванні слід пам'ятати про нормальну форму бази даних. Нормальна форма – властивість моделі даних, яка характеризує її надмірність. Додаткова надмірність може призвести до логічно не вірних результатів як вибірки так і занесення даних. Дотримання хоча б 3 нормальної форми може значно знизи навантаження на СУБД, зменшити об'єм пам'яті, що займають файли баз даних та збільшити зручність використання БД. Якщо база даних знаходиться в третій нормальній формі, це свідчить про те, що:

- кожна таблиця має свій первинний ключ;
- категорії дані, що повторюються в різних записах виносяться в інші таблиці;
- кожен атрибут повинен мати лише одне значення;
- поле, що залежить від основного ключа та іншого поля виноситься в окрему таблицю.

Варто зауважити, що не потрібно зберігати дані, якщо вони просто вираховуються з інших записів.

### 3.2.2 Вибір моделі даних

Модель даних – абстрактне представлення деяких об’єктів та зв’язків між ними, що представляють собою деяку логічну структуру.

На сьогоднішній день СУБД використовують такі моделі даних:

- Реляційна;
- Ієрархічна;
- Мережева;
- Об’єктно-орієнтована.

Реляційна модель даних є найрозповсюдженішою моделлю. В даній моделі даних використовується високий рівень абстракції та не потребує додаткових структур для організації збереження даних і не залежить від

способу їх фізичного представлення. Це досягнуто завдяки використання теорії відношень. Дані в реляційній моделі зберігаються у вигляді таблиць та зв’язків між ними. Кожен в таблиці рядок описує один об’єкт, а кожен стовпець це деяка характеристика об’єкту. Зв’язки представленні у вигляді ключів, що належать деяким стовпцям. Переваги: простота та зручність реалізації, концепція (зв’язки та таблиці) добре розуміється користувачами. Недоліки: неможливо ідентифікувати окремий запис поза таблиці, якій належить запис.

Ієрархічна модель даних розроблена на принципі дерева. Кожен об’єкт може мати лише один батьківський елемент на безліч дочірніх. Головний запис кожного дерева повинен мати унікальний ключ. Між дочірніми елементами одного шляху також має бути унікальний ключ. Кожен запис ідентифікується повним ключем – сукупність всіх ключів від головного до обраного запису. При видаленні батьківського видаляються всі дочірні елементи. Переваги: ефективне використання пам’яті серверу, велика швидкість операцій над даними. Недоліки: складність побудови та зміни схеми БД, складні для розуміння зв’язки та громіздкість.

Мережева модель даних дуже схожа на ієрархічну, проте основною відмінністю є те, що у кожного запису може бути декілька батьківських. В цій моделі, в порівнянні з ієрархічною, введено нові поняття: тип та екземпляр. Тип задається ім’ям за задає властивості екземпляру даного типу. Екземпляр представлений у вигляді запису-батьком та сукупністю дочірніх. Екземпляр запису не може належати до 2 або більше групових відношень одного типу. Переваги та недоліки повністю перейшли з ієрархічної моделі.

Об’єктно-орієнтована модель представлена у вигляді класів, об’єктів, їх атрибутів та методів. Така модель дозволяє працювати з даними як в об’єктно-орієнтованих мовах програмування. Тобто, загальна структура представлена деревом, кожне ребро це зв’язок, а вершина – об’єкт, а при побудові можна використовувати такі принципи ООП як інкапсуляція, наслідування та поліморфізм. Переваги: можна зручно використовувати з ООМП, визначати методи обробки та виводу даних, використання принципів ООП дозволяє

зручніше представляти дані. Недоліки: висока складність побудови та відносно низька швидкість виконання запитів.

Зручність реалізації та використання теорії відношень призвело до того, що реляційна модель є найпопулярнішою, однією з найшвидших та найнадійніших, стала сучасним стандартом. Саме на цій моделі працюють такі СУБД як MySQL, Oracle SQL Server та Microsoft SQL Server. Я також обрав цю модель даних для СУБД через великий асортимент СУБД, що підтримують цю модель, також зручність у розумінні моделі та СУБД, що вже працює в КБ ІС працює саме на цій моделі.

### 3.2.3 Вибір СУБД

СУБД – система управління базами даних, програмне забезпечення, що дозволяє керувати базами даних та використовувати їх. Тобто надає можливості створення, оновлення, видалення, збереження даних, забезпечує контроль доступу.

СУБД може класифікуватись за деякими ознаками:

- За моделлю даних;
- За способом доступу до БД;
- За розподіленістю.

З моделлю даних ми визначились в попередньому пункті. Розглянемо як саме класифікують СУБД за розподіленістю.

Класифікація за розподіленістю:

- Локальна;
- Розподілена.

Локальна СУБД це така СУБД, що встановлюється на одному комп’ютері. Розподілена СУБД встановлюється на декілька комп’ютерів проте може працювати як одна система.

Сучасні СУБД можуть працювати як локально так і розподілено. Проте правильне налаштування розподіленої БД може зайняти велику кількість часу. Так як сучасні СУБД можуть працювати в двох режимах, тому цим пунктом

для вибору СУБД можна знехтувати. Тепер розглянемо більш детально як саме СУБД класифікуються за способом доступу.

Класифікація за способом доступу до БД:

- Файл-серверні;
- Клієнт-серверні;
- Вбудовані.

Файл-серверні СУБД розташовуються в деякій мережі. СУБД розташована на кожному комп'ютері, а дані синхронізуються через мережу. Під час роботи з файлами, вони блокуються для користувачів. Переваги: низьке навантаження на сервер. Недоліки: відсутність централізованого керування, підвищене навантаження на мережу, ускладненість реалізації основних принципи збереження даних (надійність, доступність та безпечність).

Клієнт-серверні СУБД розташовуються на сервері баз даних та здійснює доступ та оновлення інформації за допомогою запитів. Переваги: нижче навантаження на мережу клієнтів, централізоване керування, зручно забезпечує основні принципи збереження даних (надійність, доступність та безпечність). Недоліки: сервер потрібен буди відносно потужний, велике навантаження на мережу сервера.

Вбудовані СУБД постачаються як частина іншого програмного забезпечення та не потребує встановлення. Така база даних використовується для збереження даних локально та не підходить для роботи через мережу.

Так як файл-серверна архітектура вважається застарілою, а використання вбудованої неможливе через неможливість працювати по мережі, то залишається клієнт-серверна.

Реляційних клієнт-серверних СУБД на даний момент представлено дуже багато. Серед найпопулярніших є MySQL, Oracle SQL Server та Microsoft SQL Server [21]. Вони дуже схожі проте мають деякі особливості (табл. 3.1).

Таблиця 3.1 – Порівняння особливостей БД

| Назва | MySQL | Oracle SQL Server | Microsoft SQL Server |
|---|---|---|---|
| Безкоштовне використання | Повністю безкоштовна | Має безкоштовну версію з обмеженим функціоналом | Має безкоштовну версію з обмеженим функціоналом |
| .Net data provider | ADO.NET | ODP.NET | ADO.NET |
| Підтримка SQL | SQL | Oracle SQL | Transact SQL |
| Адміністрування | phpMyAdmin | Oracle SQL Developer | Management Studio |

Так як додаток написаний на мові програмування C# то варто звернути увагу на зручність використання СУБД із цією мовою. ADO.NET вже вбудована у .Net Framework а ODP.NET через вбудовану у Visual Studio систему управління пакетами NuGet. Якщо ж для розробки використовується не Visual Studio то потрібно буде вручну додати цей пакет до додатку на налаштовувати його.

Кожна із СУБД підтримує SQL та її підмови DDL, DML и DCL. Проте Microsoft використовує свій діалект, а Oracle свій. Кожен із діалектів розширює функціонал SQL.

Налаштування та управління СУБД здійснюється через спеціально розроблені для цього додати. У MySQL це веб-додаток, а у Microsoft та Oracle це десктопні додатки. Вони мають базовий функціонал управління БД та деякий додатковий. У Oracle SQL Developer він занадто малий. Management Studio в свою чергу дає можливості, крім базовим управлінням БД, зручно

створювати таблиці мовою T-SQL або за допомогою схем, має IntelliSense T-SQL мови та ще багато зручних інструментів.

Так як Microsoft SQL Server має багато переваг у супутньому ПЗ, зручному доступі, без встановлення додаткових бібліотек, до БД через мову C# та на ньому вже були реалізовані інші проекти КБ ІС то для роботи я обрав СУБД Microsoft SQL Server.

### 3.2.4 Опис спроектованих таблиць

Під час розробки системи “Проект Документу” була спроектована база даних. Вона містить дані про роботу (основна робота, навантаження, ставки і т.д.), її зміну, інформацію про працівників. Ця база даних може використовуватись не лише для цього проекту а і для інших. Розглянемо детальніше деякі таблиці.

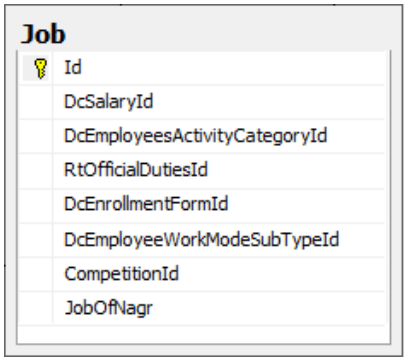


Рисунок 3.1 – Таблиця БД “Job”

Дана таблиця проміжна і поєднує основні дані про роботу працівника. За допомогою неї та поєднаних з нею таблиць можна визначити яка саме ставка у працівника, яка в нього трудова діяльність, його обов’язки.

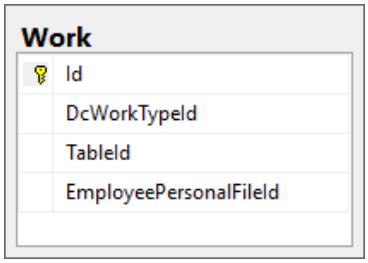


Рисунок 3.2 – Таблиця БД “Work”

Таблиця зберігає дані про трудову діяльність. Тож за допомогою допоміжної таблиці можна зрозуміти яка робота основна, а яка йде лише як

навантаження на основну. Також за допомогою допоміжних таблиць можна визначити історію даного працівника: з якої на яку роботу він переходив.

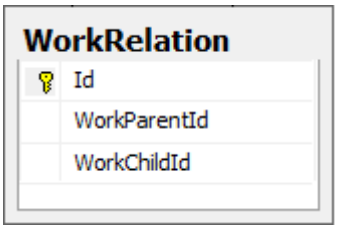


Рисунок 3.3 – Таблиця БД “WorkRelation”

Допоміжна таблиця Work. За допомогою неї можна визначити відношення між роботами: яка з них головна, а яка є навантаженням на основну.

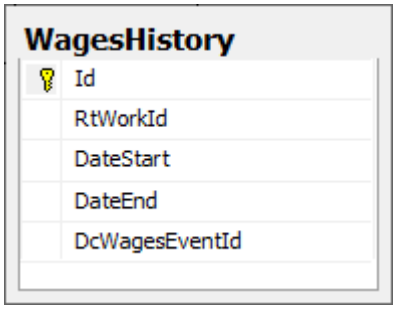


Рисунок 3.4 – Таблиця БД “WagesHistory”

Дана таблиця проміжна та за допомогою неї можна визначити коли саме встановили, змінили, подовжили чи відмінили оклад співробітника, розмір окладу, до якої ставки він належить.


Рисунок 3.5 – Таблиця БД “Employee”

Ця таблиця з’єднує основні таблиці з даними про співробітника. До цих даних відносяться:

- Посилання на фотокартку працівника;
- Дані про індивідуальний податковий номер;
- Дані про ідентифікаційні документи;

- Місце проживання;
- Дані про трудову книжку.

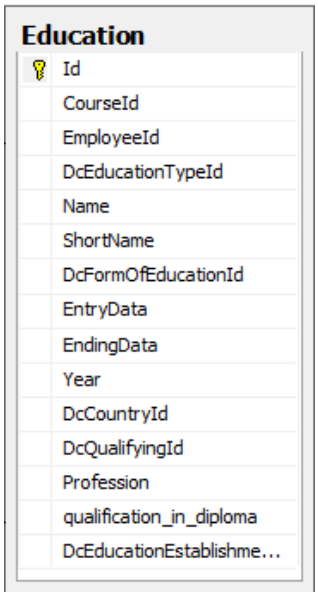


Рисунок 3.6 – Таблиця БД “Education”

В даній таблиці зберігаються основні дані про освіту чи науковий ступінь співробітника.

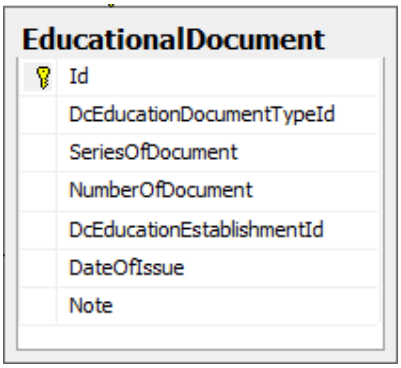


Рисунок 3.7 – Таблиця БД “EducationalDocument”

Таблиця зберігає основні дані про документи, що засвідчують освіту чи науковий ступінь.

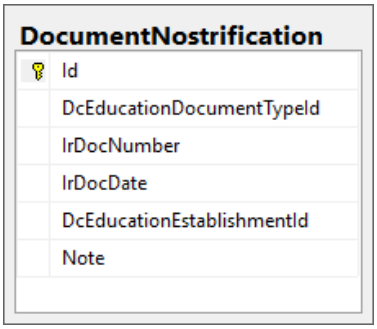


Рисунок 3.8 – Таблиця БД “DocumentNostrification”

Дана таблиця зберігає дані про документи, що нострифікують документи про іноземну освіту чи науковий ступінь.

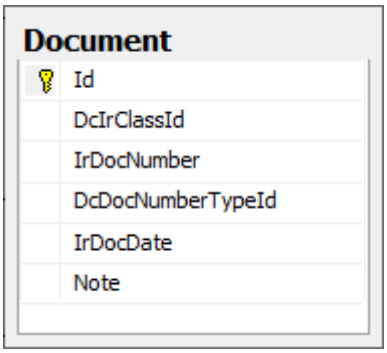


Рисунок 3.9 – Таблиця БД “Document”

В цій таблиці зберігаються дані про документ. Дані ж самого документу містяться в параграфах.

## 3.3 Проектування додатку користувача

### 3.3.1 Вибір мови програмування

На сьогоднішній день представлено безліч мов програмування. Проте найпопулярнішими, для розробки декстопних додатків, є C подібні мови. Серед них найпопулярнішими є C# та Java [20]. Щоб обрати відповідну, треба розглянути особливості, переваги та недоліки кожної з них.

C# – це мова програмування загального призначення, який вперше з'явився в 2000 році в рамках ініціативи Microsoft .NET. Він був розроблений для загальної мовної інфраструктури (CLI) – відкритої специфікації, розробленої Microsoft і стандартизованої ISO і ECMA. Додатки C# скомпільовані в байт-код, який може запускатися при реалізації CLI.

Java, спочатку випущений Sun Microsystems в 1995 році, є мовою програмування загального призначення, який був розроблений з конкретною метою, що дозволяє розробникам "write once, run anywhere", тобто написати код один раз і запускати в будь-якому місці. Java-додатка скомпільовані в байт-код, який може запускатися при реалізації віртуальної машини Java (JVM). Подібно CLI, JVM допомагає подолати розрив між вихідним кодом і 1 і 0, які розуміє комп'ютер.

Поява як Java, так і C#, тісно пов'язане з переходом від низькорівневих мов програмування, таких як мови програмування C++, до мов вищого рівня, які компілюються в байт-код. Байт-код можна запустити на віртуальній машині. З цим пов'язаний ряд переваг, в першу чергу, можливість написання коду, який буде зрозумілий людині і буде працювати на будь-якій апаратній архітектурі, на якій встановлена віртуальна машина. Якщо відкинути синтаксичні примхи в сторону, то не дивно, що ці два подібні між собою мови так популярні для розробників додатків. Ось кілька основних подібностей між C # і Java:

Безпека типів. Помилка типу виникає, коли тип даних одного об'єкта помилково призначається іншому об'єкту, створюючи ненавмисні побічні ефекти. І C#, і Java працюють так, щоб гарантувати виявлення таких типів привидів під час компіляції. Якщо приведення не може бути застосоване до нового типу, тоді під час виконання виникне виключення (exception).

Прибирання сміття: На мовах нижчого рівня управління пам'яттю може бути складним, адже потрібно пам'ятати про те, що необхідно правильно видалити нові об'єкти, щоб звільнити ресурси. На C # і Java є вбудована прибирання сміття, яка допомагає запобігти витоку пам'яті шляхом видалення об'єктів, які більше не використовуються додатком. Витоки пам'яті все ще можуть виникати, але завдяки автоматичному управлінню пам'яттю – це вже не наша проблема.

Одиночне спадкоємство (Single inheritance). Обидві мови підтримують одиночне спадкоємство – це означає, що існує тільки один шлях з будь-якого базового класу в будь-який з його похідних класів. Це обмежує ненавмисні побічні ефекти, які можуть виникати при наявності декількох шляхів між декількома базовими класами і похідними класами. Diamond pattern – книжковий приклад цієї проблеми.

Інтерфейси. Інтерфейс являє собою абстрактний клас, де всі методи абстрактні. Абстрактним методом є той метод, який оголошений, але не містить подробиць його реалізації. Код, що визначає будь-які методи або властивості, певні інтерфейсом, має надаватися класом, який його реалізує. Це допомагає

уникнути двозначності паттерна diamond, оскільки завжди ясно, який базовий клас реалізує даний похідний клас під час виконання. Результатом є чиста ієрархія лінійних класів одиночного наслідування в поєднанні з деякою універсальністю множинного спадкоємства. Фактично використання абстрактних класів є одним із способів множинного успадкування мов, які можуть подолати проблему паттерна diamond.

Важливо пам'ятати, що C # бере свій початок в бажанні Microsoft мати власний «Java-подібний» мову для платформи .NET. Оскільки C# не створювалася в вакуумі, нові функції були додані і налаштовані для вирішення проблем, з якими стикалися розробники Microsoft, коли вони спочатку намагалися створити свою платформу на Visual J++. У той же час співтовариство Java з відкритим вихідним кодом продовжувала зростати і між цими двома мовами розвивалася гонка технічних озброєнь. Ось деякі з основних відмінностей між C # і Java.

Windows vs open-source. Хоча існують реалізації з відкритим вихідним кодом, C# в основному використовується в розробці для платформ Microsoft – .NET Framework CLR і є найбільш широко використовуваної реалізацією CLI. На іншому кінці спектру Java має величезну екосистему з відкритим вихідним кодом і у нього відкрилося друге дихання частково завдяки тому, що Google використовує JVM для Android.

Підтримка узагальнень (Generics): Generics покращує перевірку типів за допомогою компілятора, в основному видаляючи приведення з вихідного коду. В Java узагальненя реалізуються з використанням стирань. Параметри загального типу «стираються», а при компіляції в байт-код додаються приведення. C# також використовує узагальнення, інтегруючи його в CLI і надаючи інформацію про тип під час виконання, що дає невелике збільшення продуктивності.

Підтримка делегатів (покажчиків): У C# є делегати, які по суті є покажчиками на методи. Для досягнення такої ж функціональності в Java вам необхідно використовувати інтерфейс з одним методом або іншим способом

обходу, який може зажадати нетривіального кількості додаткового коду, в залежності від програми.

Виключення, що перевіряються: Java розрізняє два типи винятків – ті що перевіряються і не перевіряються. C # вибрав більш мінімалістський підхід, маючи тільки один тип винятку. Хоча здатність ловити виключення може бути корисна, вона також може мати негативний вплив на масштабованість і контроль версій.

Поліморфізм: C # і Java використовують дуже різні підходи до поліморфізму. Java допускає поліморфізм за замовчуванням, C# же повинен викликати ключове слово «virtual» в базовому класі і ключове слово «override» в похідному класі.

Перерахування (Enums): в C# перерахування представляють собою прості списки іменованих констант, де базовий тип повинен бути цілим числом. Java являє перерахування більш глибоко, розглядаючи його як іменований екземпляр типу, що спрощує додавання користувацького поведінки окремих перерахуванням.

### 3.3.2 Додаткові контроли

Для потреб додатку було не достатньо використовувати лише стандартні контроли, тож було розроблено декілька контролів із розширеним функціоналом.

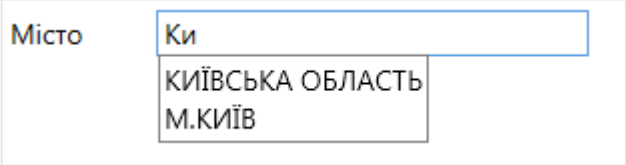


Рисунок 3.10 – Поле для текстового вводу із випливаючою підказкою

TextBox_AutoComplete – контрол із випливаючою підказкою (рис. 3.10). Під час роботи над проектом, потрібно було використовувати контрол, що має автозаповнення, під час того, як користувач друкує. Для цього можна було скористатися контролом типу ComboBox, проте було б важко налаштувати його вигляд так, щоб він виглядав як звичайний TextBox. І коли б завантажувались

нові записи, було б не можливо показати користувачеві, що ще йде пошук збігу за вводом. Тож було вирішено написати новий контрол.

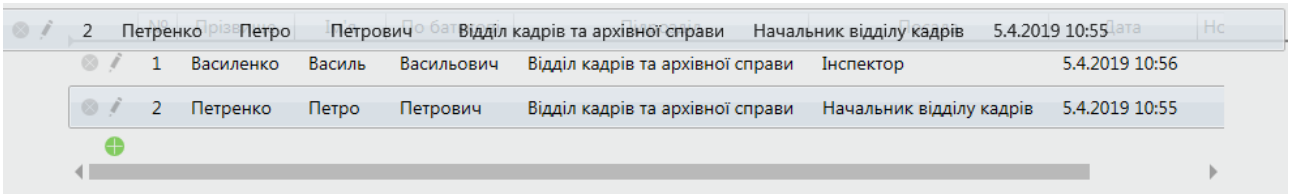


Рисунок 3.11 – Модифікований контрол-список

ListView_Custom сильно модифікований ListView, який використовується для зручного та зрозумілого подання таблиць (рис. 3.11). Змінений завдяки реалізації додаткового функціоналу та зміні зовнішнього вигляду за допомогою редагування його ControlTemplate та DataTemplate. Додатковий функціонал:

- Автоматичне підлаштування ширини колонок під елементи, підлаштування загальної ширини;
- Інтуїтивно-зрозумілі кнопки додавання, редагування та видалення елементів списку;
- Автоматично вказує номер запису в таблиці та зберігає його для кожного запису;
- Drag&Drop можна сортувати записи в таблиці перетягуванням цих записів, номер автоматично перераховується;
- Змінений зовнішній вигляд, використані різні Converter та DataTemplates для збільшення функціоналу з візуальної точки зору. Наприклад вибір елементів у моделі буде представлений IEnumerable<T> у моделі на ComboBox у візуальній частині.

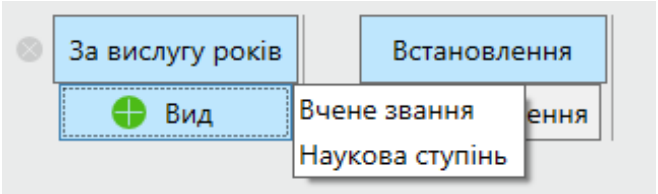


Рисунок 3.12 – Модифікований ComboBox

*ComboBox_WidthAutoAdjust* автоматично підлаштовуватись під найширший елемент. Зручно використовувати в *ListView_Custom*, для автоматичного підлаштування колонок таблиці під найширші елементи.

*ComboBox_SelectionMenu* виглядає як звичайна кнопка (рис. 3.12). При натисканні, поруч з'являється список, а після вибору обраний елемент не заноситься в ContentPresenter, тобто працює як звичайна кнопка, проте створює подію із обраним елементом (треба підписатись на подію SelectionChanged). Використовується для меню, наприклад, для зручного вибору виду надбавок.

RadioButton_MenuItem повторює функціонал звичайної RadioButton проте модифікована для роботи з меню. Був переписаний зовнішній вигляд та додана кнопка для можливості видалення обраного елементу меню.

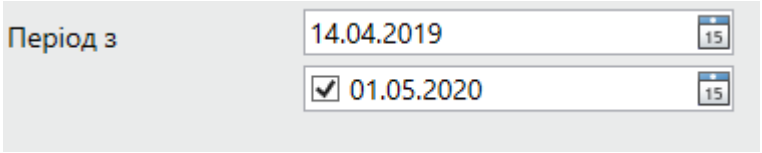


Рисунок 3.13 – Модифікований DatePicker

DatePicker_Custom був переписаний стандартний дизайн самого контролу та доданий CheckBox для зручної роботи з проміжками дат (рис. 3.13). Якщо кінцева дата не визначена, то CheckBox “вимикає” контрол. Якщо потрібно вказати кінцеву дату, то потрібно натиснути на CheckBox та “ввімкнути контрол”.

### 3.3.3 Проектування та розробка додатку

Патерн MVVM (Model-View-ViewModel) дозволяє відокремити логіку додатки від візуальної частини (подання). Даний патерн є архітектурним, тобто він задає загальну архітектуру програми. Даний патерн був представлений Джоном Госсманом (John Gossman) в 2005 році як модифікація шаблону Presentation Model [4] і був спочатку націлений на розробку додатків в WPF. І хоча зараз цей патерн вийшов за межі WPF і застосовується в самих різних технологіях, в тому числі при розробці під Android, iOS, проте WPF є досить показовою технологією, яка розкриває можливості даного патерну. MVVM складається з трьох компонентів: Model, ViewModel і View.

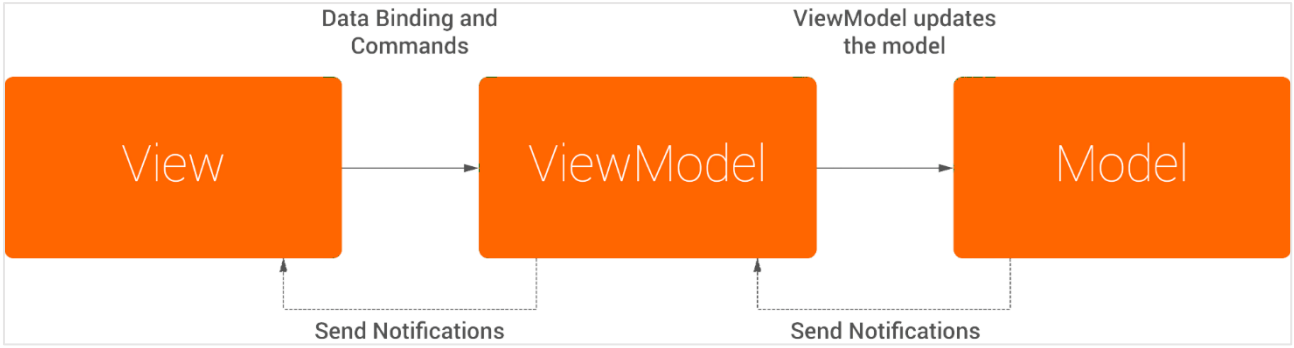


Рисунок 3.14 – Схема патерну MVVM

• Model описує використовувані в додатку дані (рис 3.14). Моделі можуть містити логіку, безпосередньо пов'язану цими даними, наприклад, логіку валідації властивостей моделі. У той же час модель не повинна містити ніякої логіки, пов'язаної з відображенням даних, взаємодією з візуальними елементами управління та бізнес-логіки. Нерідко модель реалізує інтерфейси INotifyPropertyChanged або INotifyCollectionChanged, які дозволяють повідомляти систему про зміни властивостей моделі. Завдяки цьому полегшується прив'язка до View, хоча знову ж пряму взаємодію між моделлю і представленням відсутня.

• View визначає візуальний інтерфейс, через який користувач взаємодіє з додатком (рис 3.14). Стосовно до WPF – це код в XAML, який визначає інтерфейс у вигляді кнопок, текстових полів та інших візуальних елементів. Хоча вікно (клас Window) в WPF може містити як інтерфейс в XAML, так і прив'язаний до нього код C #, проте в ідеалі код C # не повинен містити якийсь логіки, крім хіба що конструктора, який викликає метод InitializeComponent і виконує початкову ініціалізацію вікна, та логіки що відноситься лише до View. Вся ж основна логіка додатку виноситься в компонент ViewModel. Однак іноді в файлі пов'язаного коду все може знаходитися певна логіка, яку важко реалізувати в рамках паттерна MVVM у ViewModel. View не використовує події (Events) за рідкісним винятком, а виконує дії (Actions) в основному за допомогою команд.

• ViewModel пов'язує Model і View через механізм прив'язки даних (Bindings) (рис 3.14). Якщо в моделі змінюються значення властивостей, при реалізації моделлю інтерфейсу INotifyPropertyChanged автоматично йде зміна

відображуваних даних в поданні, хоча безпосередньо модель і уявлення не пов'язані. ViewModel також містить логіку по отриманню даних з моделі, які потім передаються в уявлення. І також ViewModel визначає логіку по оновленню даних в моделі. Оскільки елементи уявлення, тобто візуальні компоненти типу кнопок, не використовують події, то уявлення взаємодіє з ViewModel за допомогою команд. Наприклад, користувач хоче зберегти введені в текстове поле дані. Він натискає на кнопку і тим самим відправляє команду у ViewModel. А ViewModel вже отримує передані дані і відповідно до них оновлює модель.

Підсумком застосування патерну MVVM є функціональний розподіл програми на три компонента, які простіше розробляти і тестувати, а також в подальшому модифікувати і підтримувати.

Розробка додатку почалась з визначень моделей, які повинні бути на відображеннях. Так як для корректної роботи MVVM потрібна реалізація INotifyPropertChanged для всіх класів моделей та кожна модель повинна мати ту чи іншу валідацію даних та відображення результатів валідації на View був розроблений базовий клас для всіх моделей. Під час розробки інтерфейсу ми зіткнулися з проблемою, що під час редагування запису в будь-якій таблиці, потрібно зберігати повний стан поточної моделі, так як користувач може відмінити зміни або поточний стан моделі може не пройти валідацію. Тож щоб не зберігати кожне поле запису для кожної моделі, була додана серіалізація в пам’ять та стиснення цієї області пам’яті за допомогою Deflate для зручності та якнайменшої займаної пам’яті.

Серед стандартних методів стиснення в .Net э два методи: GZip та Deflate. В .Net вони відрізняються лише тим, що GZip додає CRC для перевірки контрольної суми, а Deflate ні. Тож очевидно що Deflate буде працювати значно швидше та стиснені дані займатимуть трохи менше пам’яті.

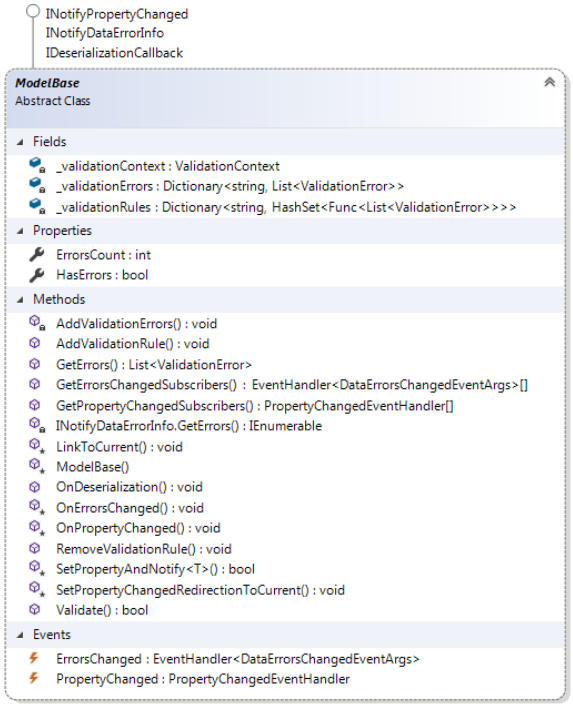


Рисунок 3.15 – Базовий клас моделей

Після створення базової моделі, були розроблені всі потрібні моделі та визначена валідація цих моделей. Базова валідація (обов'якові поля, як два поля спів-відносяться між собою (одна дата повинна бути більше за іншу або одне з двох полів має бути заповнене а інше ні)) була розроблена за допомогою механізму DataAnnotations. Він дозволяє зробити валідацію моделі за допомогою атрибутів, що значно зменшує кількість коду та покращує його читаємість [25]. Також в .Net представлений широкий вибір атрибутів валідації. Також була додана універсальна підтримка додаткової валідації за допомогою делагатів. Так як делегат в .Net це безпечний покажчик на метод, то можна додати будь-який метод до конвеєра валідації, головне щоб він відповідав сигнатурі Func<List<ValidationError>>.

Коли були розроблені механізми валідації, щоб представити користувачу результати валідації, в базовій моделі був реалізований інтерфейс INotifyDataErrorInfo, що дозволяє нативно, в рамках MVVM, передавати контролам на View результати валідації.

Було проаналізовано, як саме повинен виглядати інтерфейс додатку, тож наступним кроком, була розробка потрібних View та ViewModel, проміжної ланки між View та Model, що містять в собі потрібні моделі даних, додаткові

валідаціїї та бізнес-логіку. Так як ViewModels повинні мати широкий спектр задач, від управління меню, до форми редагування записів в списку\таблиці, були спроектовані та розроблені базові ViewModels, що відповідають певній задачі (наприклад ViewModel_Menu<T> що керує іншими ViewModels, що представляють собою єлементи меню; ViewModel_EditMasterDetails<T, U>, що має функціонал з редагування основоного запису для всього списку та редагування елементів списку).

Так як компоненти додатку за MVVM слабо зв’язані, то Views розроблялись парарельно з ViewModels. Всі Views мають лише XAML розмітку контролів, де дані отримуються та відправляються за допомогою прив’язок, та як саме показувати помилки користувачу. Вони не мають C# коду, так як View є лише відображенням ViewModel.

Тож патерн MVVM важко реалізувати лише на початку проекту, проте завдяки його перевагам, особливо слабкої зв’язаності всіх його трьох компонентів, його леко підтримувати, тобто в додатку набагато легше щось змінювати (ніж в MVP/MVC або написання додатку взагалі без патернів проектування та розробки).

Також задля кращої універсальності був використаний прицнип Inversion Of Control. Його суть полягає в тому, що кожен компонент системи повинен бути якомога більш ізольованим від інших, не покладаючись в своїй роботі на деталі конкретної реалізації інших компонентів [2]. Inversion Of Control був досягнутий за допомогою сервісів (в даному випадку сервіс – додатковий прошарок в архітектурі додатку, який виконує певну кількість вузько визначених задач) та Dependecy Injection – однііє з реалізацій IOC [24]. Dependecy Injection полягає в тому, що існують інтерфейси (найвища форма абстракції в .Net), що визначають поведінку сервісів та класи, що реалізують ці інтерфейси. В коді, де використовуються сервіси, потрібно задавати тип змінних лише інтерфейсів. Це дозволяє дуже гнучко змінювати функіонал додатку, створивши нову реалізацію інтерфейсу та лише в одному місці присвоїти обраному інтерфейсу сервіса цей єкземпляр класу.

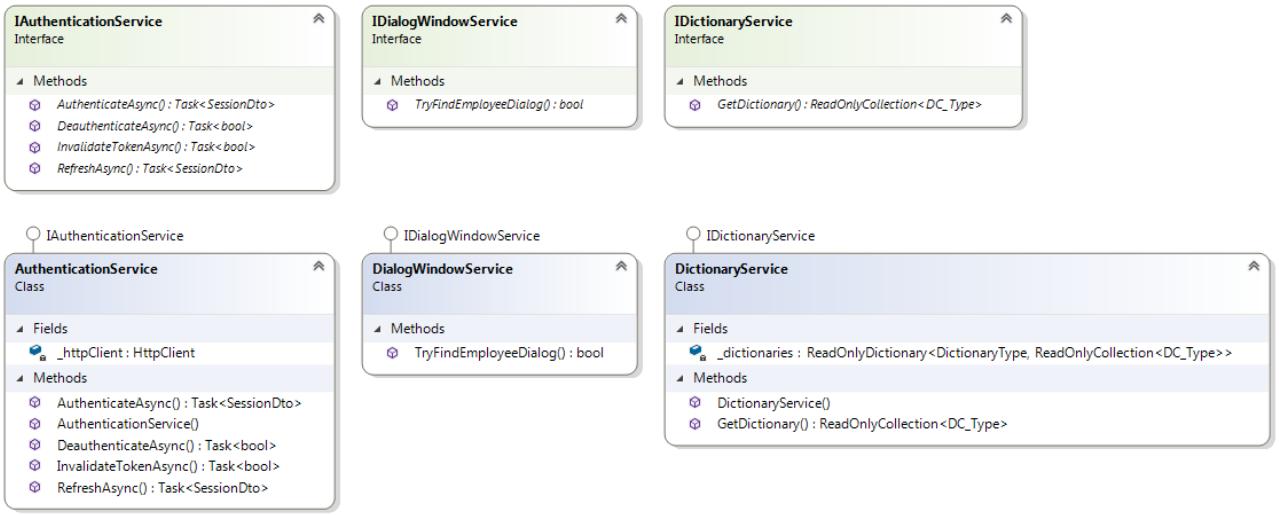


Рисунок 3.16 – Сервіси додатку

Ще однією перевагою MVVM є те, що можна використовувати мінімальну кількість вікон в додатку та легко змінювати наповнення (View) всього вікна змінюючи його DataContext. Тож було розроблено мінімальну потрібну кількість вікон.

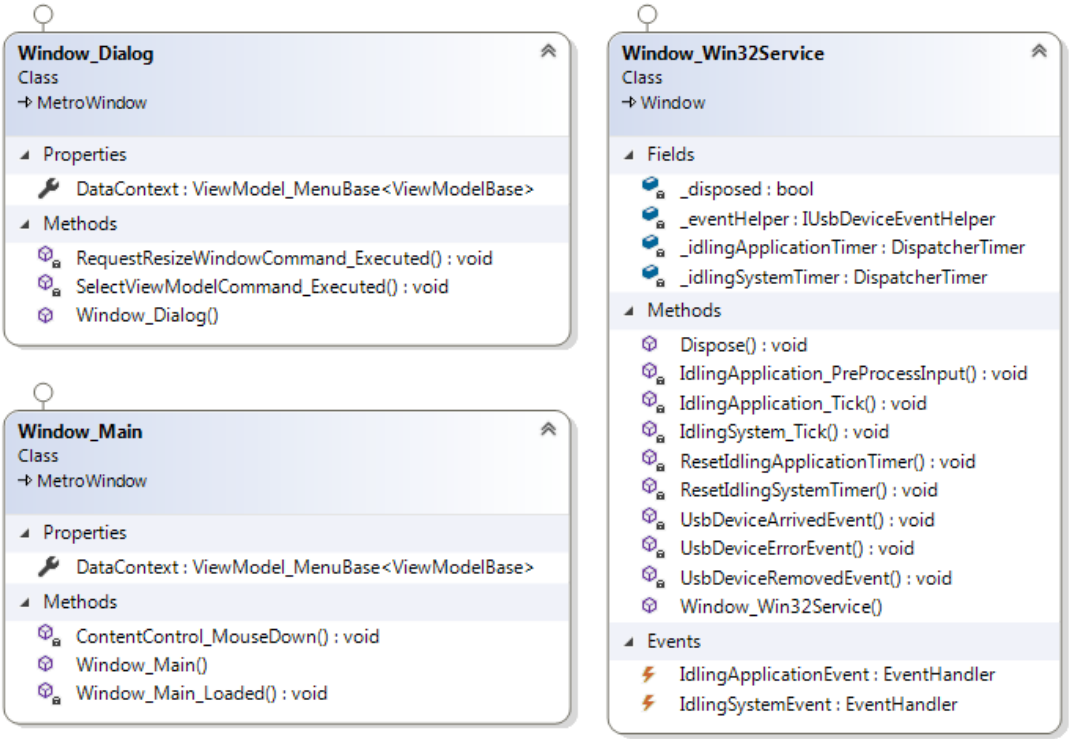


Рисунок 3.17 – Вікна додатку

Window_Main представляє собою головне вікно, а Window_Dialog – допоміжне. З точки зору функціоналу додатку, вони відрізняються лише тим, що в головному вікні відображуються повідомлення (нотифікації) та поточний користувач, а в допоміжному – ні. Проте допоміжне вікно приймає команди на

зміну розміру вікна та вибір ViewModel, який і змінить в цьому вікні View (так як завдяки MVVM View відображає до ViewModel).

Window_Win32Service допоміжне вікно іншого виду. Воно невидиме та служить для роботи з Windows Win32 Api. Завдяки Win32 Api ми можемо отримати інформацію про те, як давно система та додаток знаходяться в стані бездіяльності. Це дає змогу отримати необхідну інформацію, щоб заблокувати додаток, якщо користувач відійшов від робочого місця, що зменшить вірогідність отримання доступу до додатку (та цінних даних) стороннім людям. Також, дане вікно можна зареєструвати в Windows, щоб отримувати події, коли був під'єднаний та від'єднаний заданий з'ємний носій.

Для підвищення захисту, був розроблений функціонал ключ-флешки. За допомогою Win32 Api ми можемо читати та записувати данні на будь-який з'ємний носій в будь-яку частину цього носія. Так як найрозповсюдженішою розміткой з'ємних носіїв є FAT32, то в них перший кластер містить лише сектор завантажування [3]. В залежності від налаштувань, в FAT32 може бути від 1 до 128 секторів, тож, якщо секторів більше 1, то вони не використовуються. Так як Windows взагалі не працює з цими секторами, то данні на них в безпеці – найрозповсюдженіші віруси працюють лише з файлами та файловою розміткою (а ці сектора лежать між ними) то ж вони не зможуть добратись до даних, що там записані. Вони можуть бути знищені лише при повному форматуванні або поломці носія.

### 3.4 Вимоги до програмного та апаратного забезпечення користувача

АІС "ПД" створена за допомогою компонентів .Net Framework 4.8 на мові програмування C# та за допомогою Microsoft SQL Server. .Net Framework 4.8 потребує операційну систему не старішу за Windows 7  та деякі компоненти додатку потребують компонентів Windows, які присутні у Windows 7 та новіших версіях операційної системи.

Опираючись на ці дані ми можемо визначити мінімальні вимоги до програмного та апаратного забезпечення.

Таблиця 3.2 – Вимоги до обладнання

| **Вимоги до обладнання** | |
|---|---|
| **Процесор** | |
| Мінімум | 1 ГГц |
| **ОЗУ** | |
| Мінімум | 512 Мб |

Таблиця 3.3 – Вимоги до обладнання. Місце на диску

| **Місце на диску** | |
|---|---|
| 32-розрядна система | 4.5 Гб |
| 64-розрядна система | 4.5 Гб |

Таблиця 3.4 – Вимоги до операційної системи

| **Вимоги до операційної системи** | |
|---|---|
| Клієнтська операційна система | Windows 7 |
| Серверна операційна система | Windows Server 2008 |

Таблиця 3.5 – Вимоги до додаткового програмного забезпечення

| **Вимоги до додаткового програмного забезпечення** | |
|---|---|
| .Net Framework | 4.8 |
| Microsoft SQL Server | SQL Server 2014 |

## Висновки до розділу

У цьому розділі було розглянуто та обрано основні технології для розробки системи. Для розробки БД було обрано MsSQL Server, а для розробки клієнтського додатку та серверу – платформу .Net та мову програмування C#. Після вибору технологій була спроектована структура бази даних та розроблені спеціалізовані контроли для потреб додатку. Вибір технологій дав можливість визначити мінімальні вимоги для обладнання користувача та перейти до створення інтерфейсу користувача та тестування додатку.

# РОЗДІЛ 4 ГРАФІЧНИЙ ІНТЕРФЕЙС ТА ТЕСТУВАННЯ ДОДАТКУ

## 4.1 Графічний інтерфейс

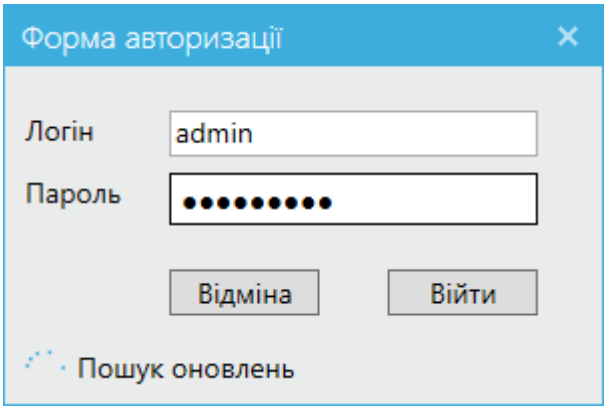


Рисунок 4.1 – Форма авторизації

Під час роботи, першу форму, що бачить користувач – форма авторизації в систему. На цій формі потрібно вказати потрібні дані для входу.


Рисунок 4.2 – Головне меню додатку

Перша форма, яку бачить користувач, після входу в систему. В головному меню можна обрати дії над документами: створити, знайти або отримати XLSX файл для повідомлення про прийняття працівника на роботу. На даний момент форми пошуку документу та отримання  повідомлення про прийняття на роботу в розробці.

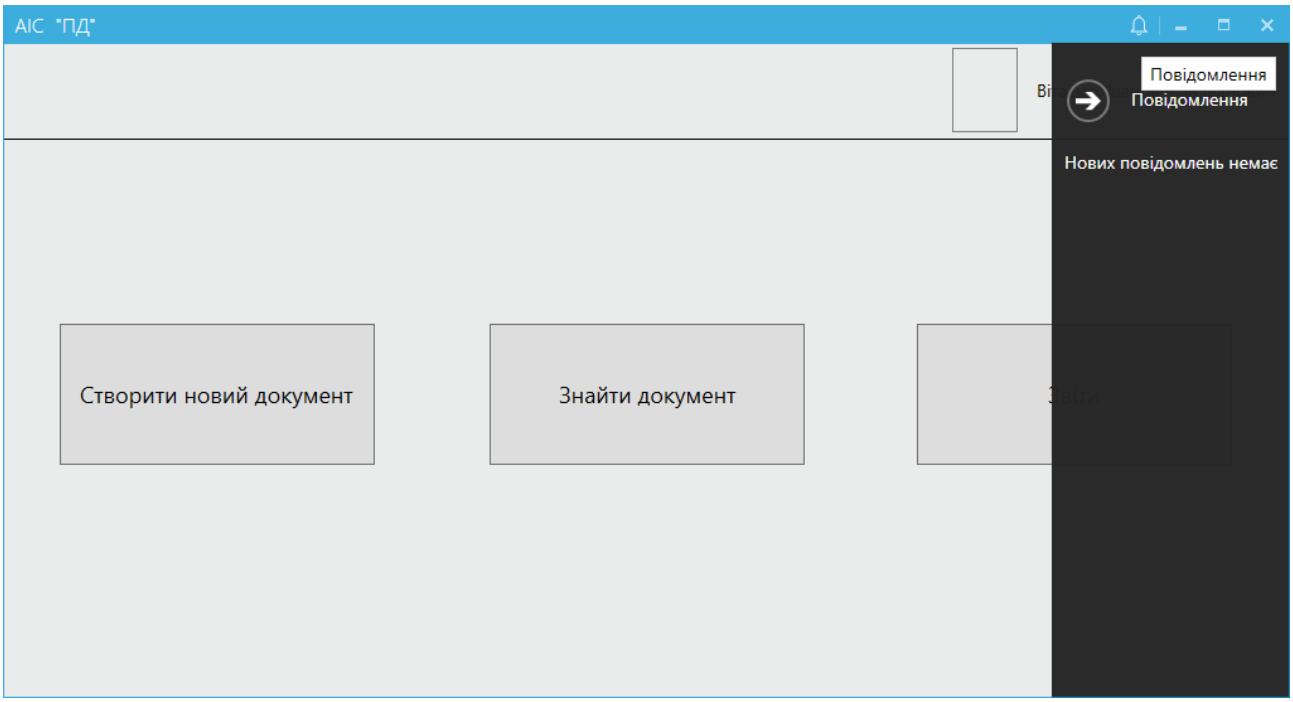


Рисунок 4.3 – Повідомлення користувачу

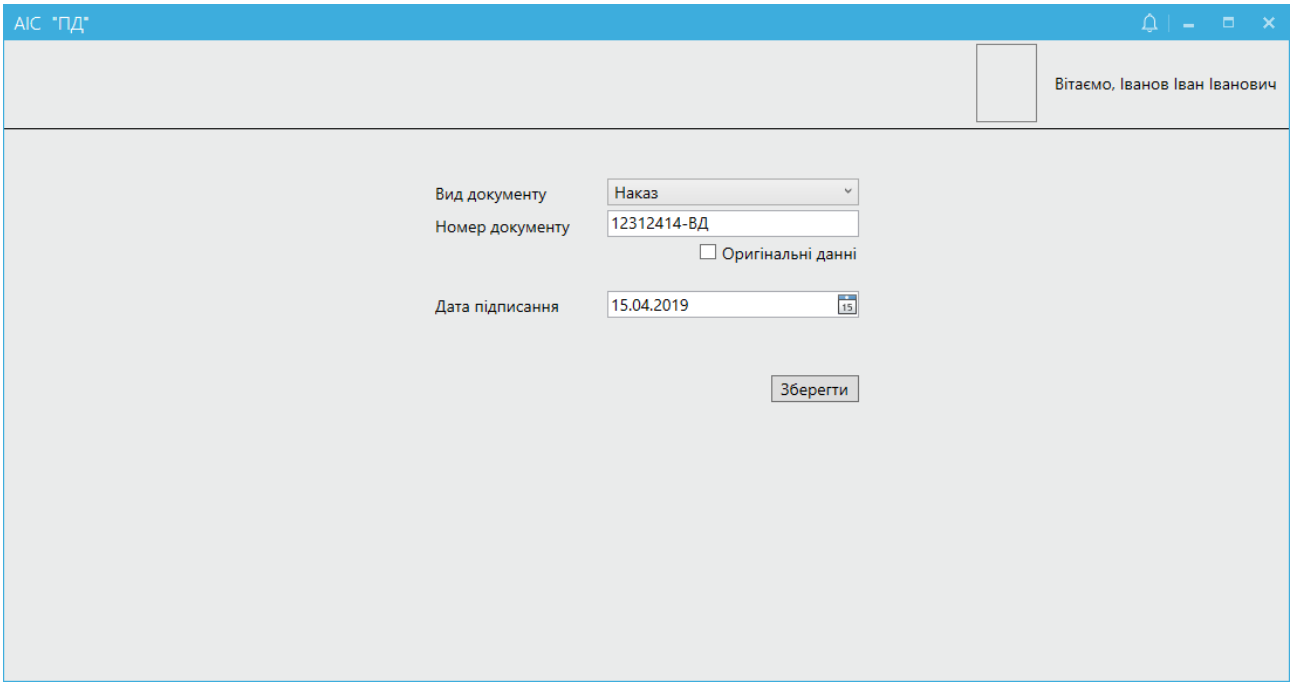


Рисунок 4.4 – Форма вибору типу, номеру та дати підписання документа

При створенні документу, перший крок, який повинен виконати користувач – вказати основні дані документа: вид, номер та дата підписання документа. Якщо номер документу невідомо, то можна і не вказувати його. Система автоматично вирахує потрібний номер. Коли відомо номер документу, треба вказати, що введений номер – оригінальні дані.

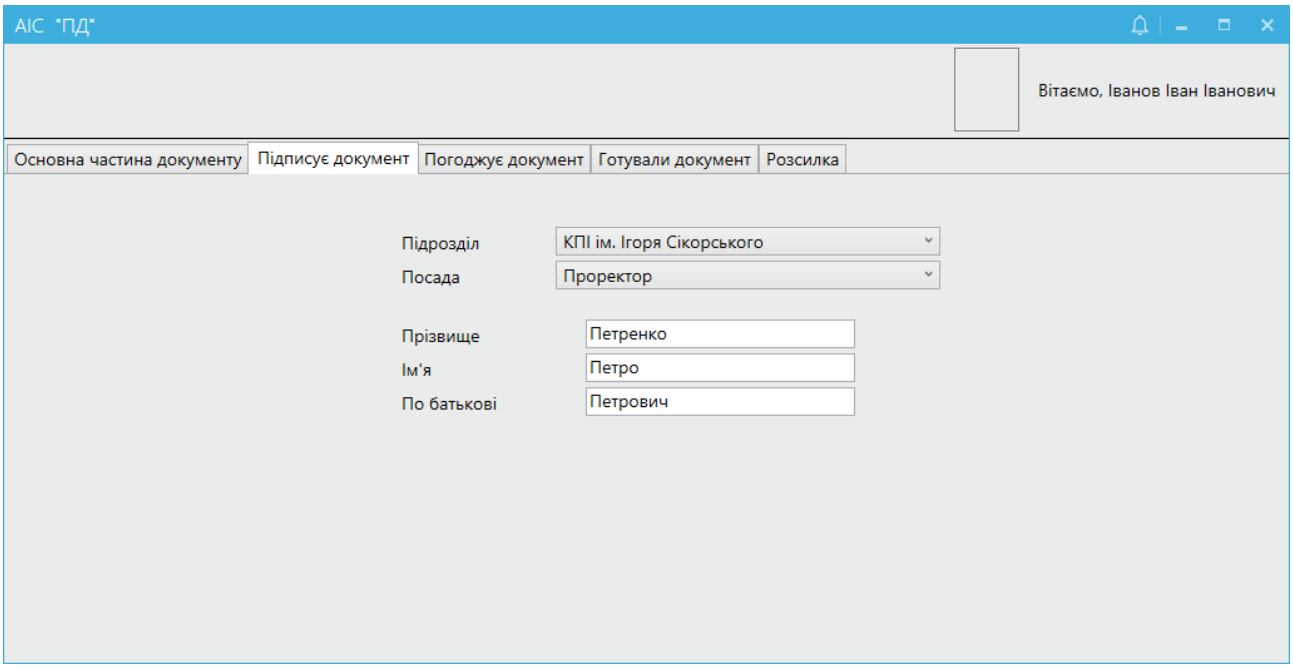


Рисунок 4.5 – Форма створення документу. Елемент меню “Підписує документ”

В цій вкладці вказується співробітник, який буде підписувати створюваний документ.

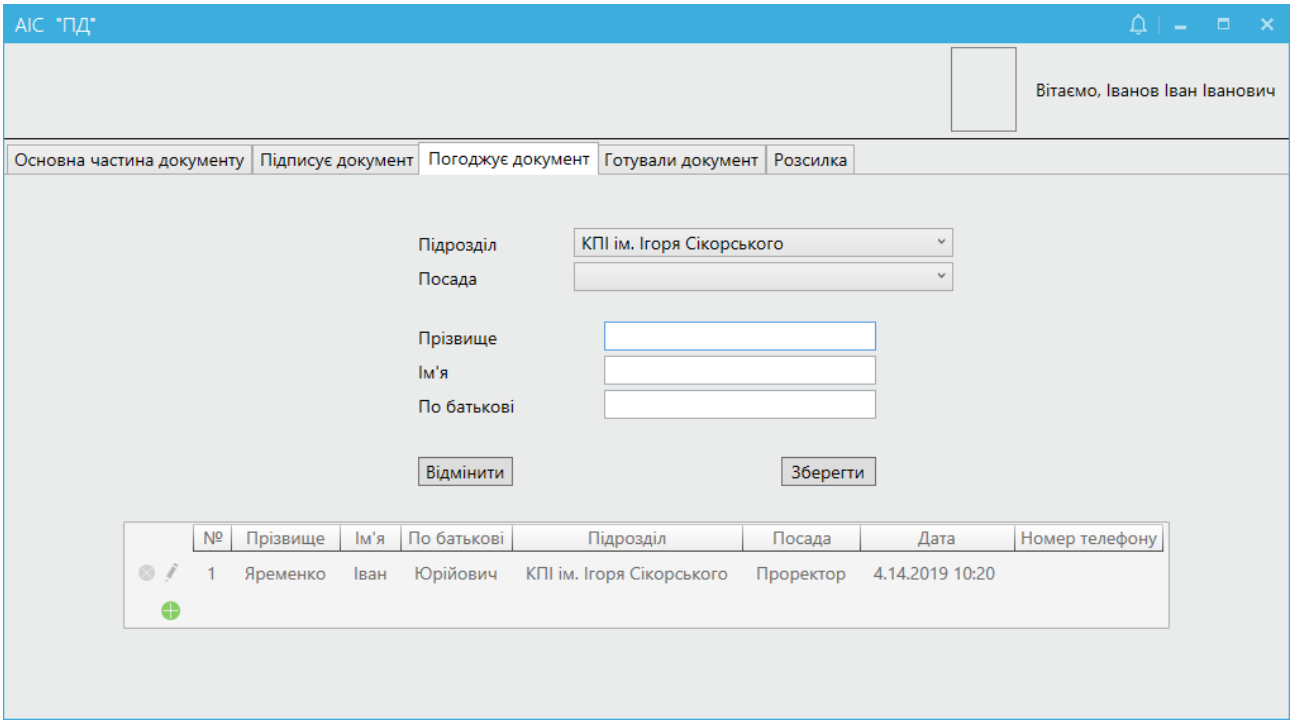


Рисунок 4.6 – Форма створення документу. Елемент меню “Погоджує документ”

В даному елементі меню заповнюються люди, що відповідальні за погодження створюваного документу. Варто зазначити, що телефонні номери

автоматично завантажуються з БД, а якщо номер телефону не вказаний або не дійсний – є можливість вказати вручну.

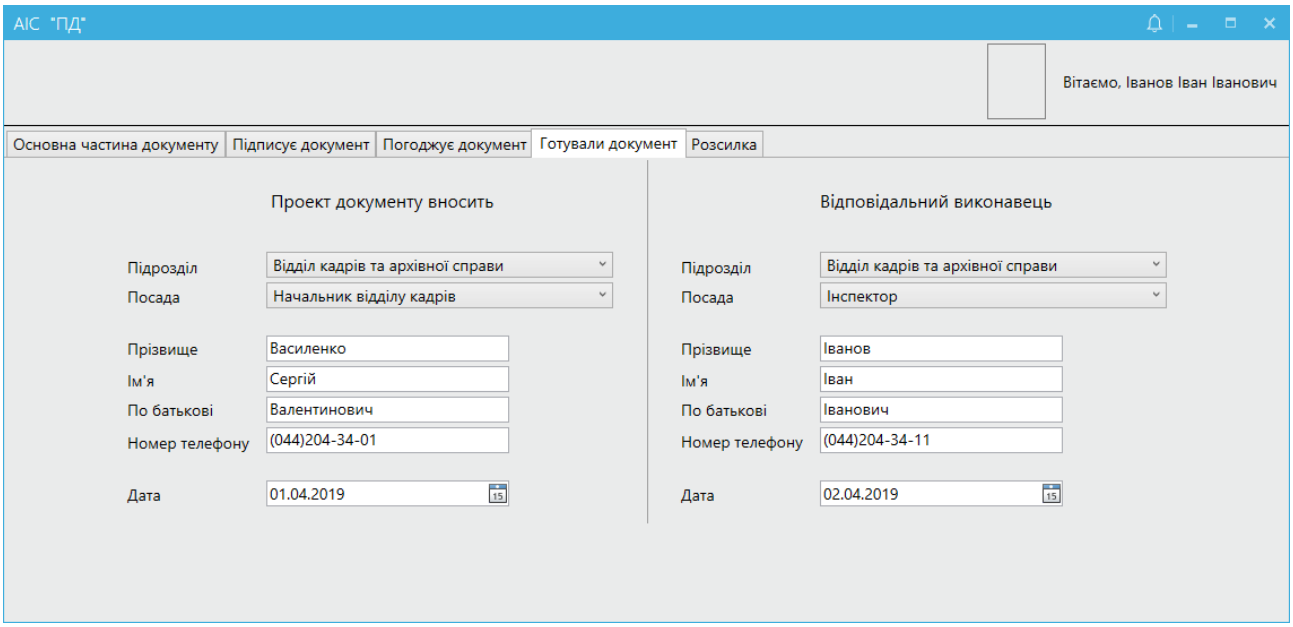


Рисунок 4.7 – Форма створення документу. Елемент меню “Документи готували”

На цій вкладці зазначаються люди, які саме готували документ та дату створення документа. Найчастіше, відповідальний виконавець – інспектор, що зараз працює в системі та , по замовчуванню, в ці поля завантажуються його дані. Проте є можливість змінити відповідальну людину.

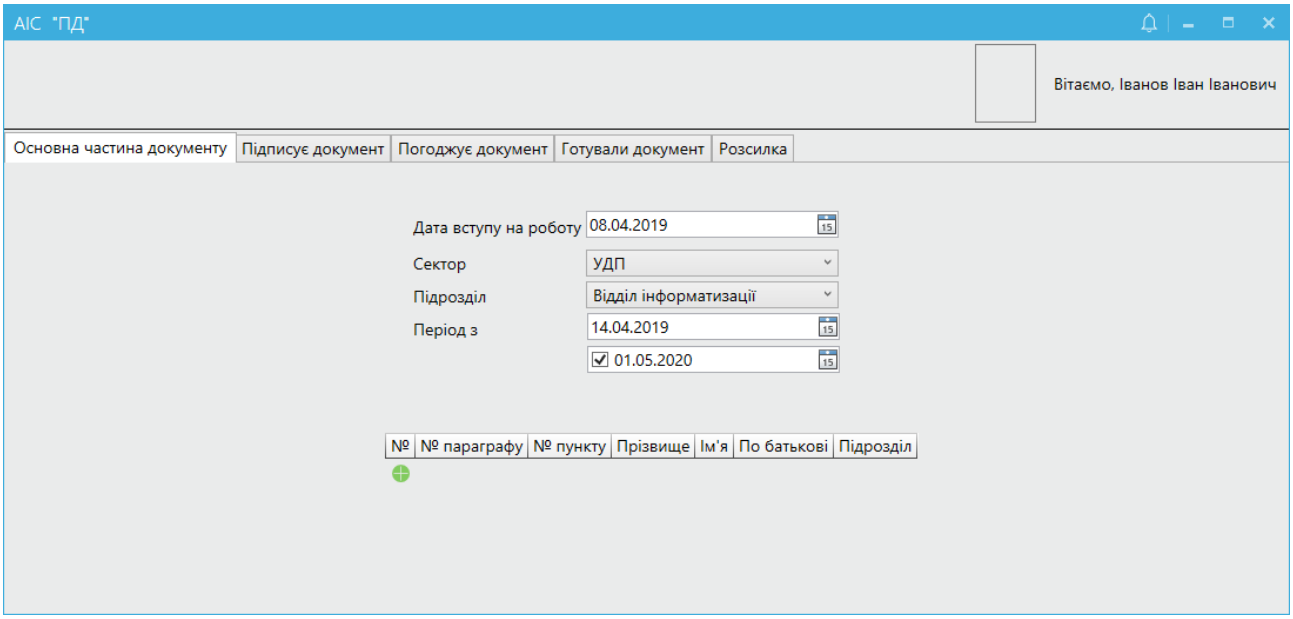


Рисунок 4.8 – Форма створення документу. Елемент меню “Основна частина документу”

Ця вкладка – основна частина документу. Тут задаються люди, які приймаються на роботу, дату вступу на роботу, період їх роботи. Кожна людина зазначається в окремому параграфі та в окремому пункті параграфу.

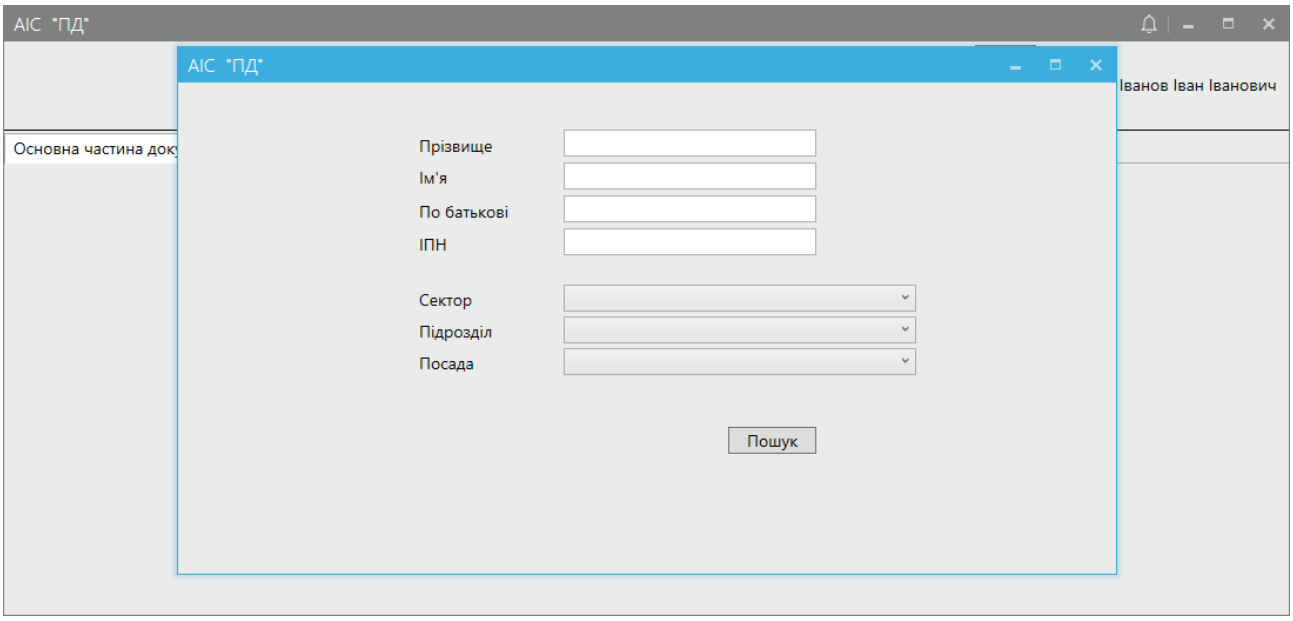


Рисунок 4.9 – Форма пошуку співробітника

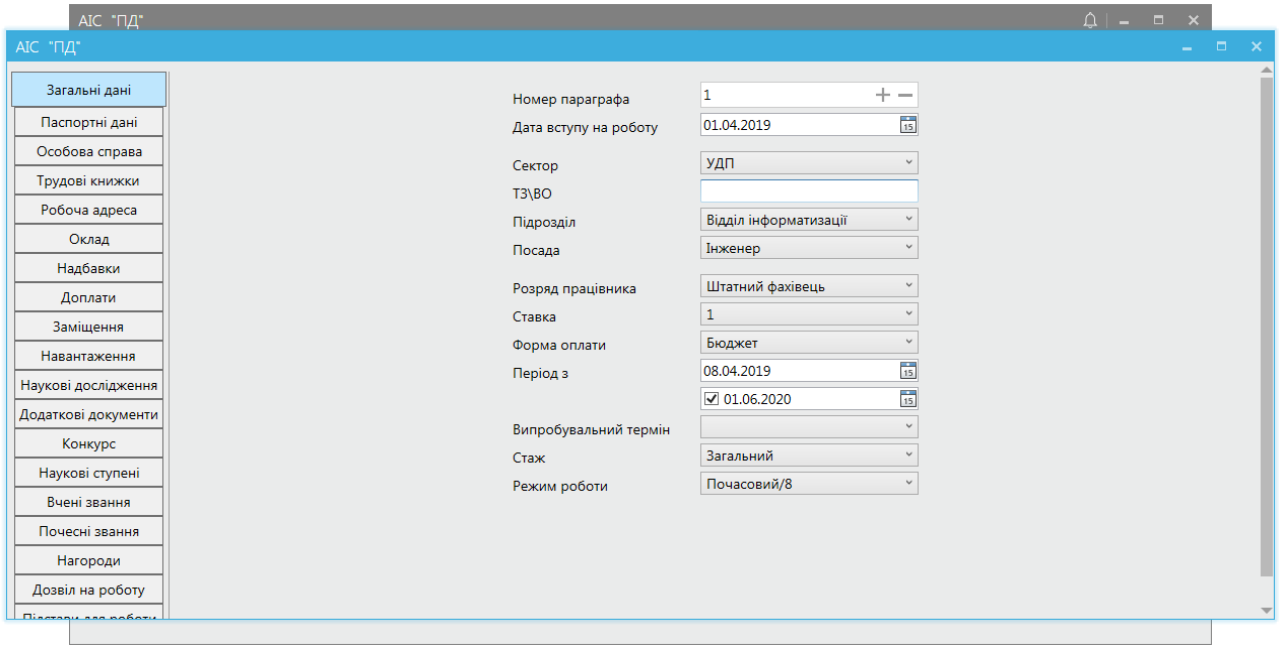


Рисунок 4.10 – Форма введення даних про працівника. Загальні дані

При додаванні нової людини до документу інспектор повинен вказати дані про людину. Якщо в університеті вже є особова справа цієї людини, то йому варто вказати її ІПН або номер особової справи. На даній вкладці задаються основні дані про особу.

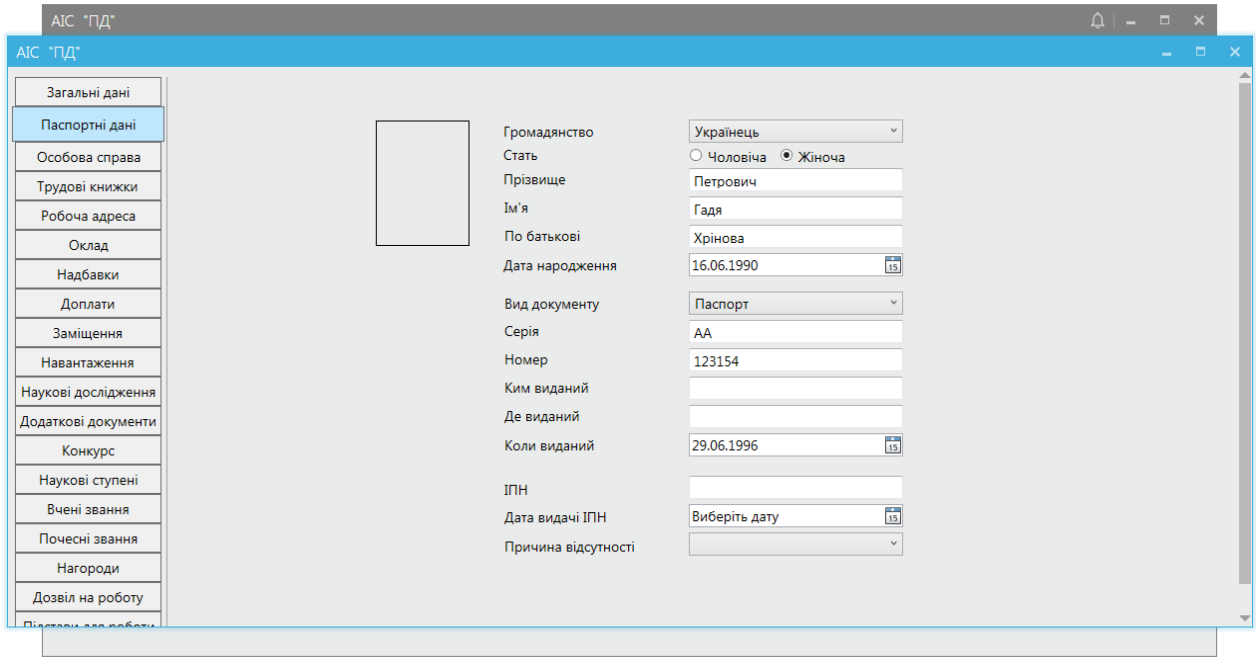


Рисунок 4.11 – Форма введення даних про працівника. Паспортні дані

В цій вкладці задаються паспортні дані особи, що приймаються на роботу. Тут потрібно вказати ІПН, дату народження та ПІБ. Особливість цієї форми в тому, що тут можлива перевірка правильності вводу ІПН, встановлення відповідності ІПН окремій особі.

## 4.2 Тестування додатку

При створенні додатку слід пам’ятати, що користувач може вводити не правильні дані або під час деяких запитів може виникнути помилка. В цьому розділі будуть наведені деякі приклади тестування додатку.

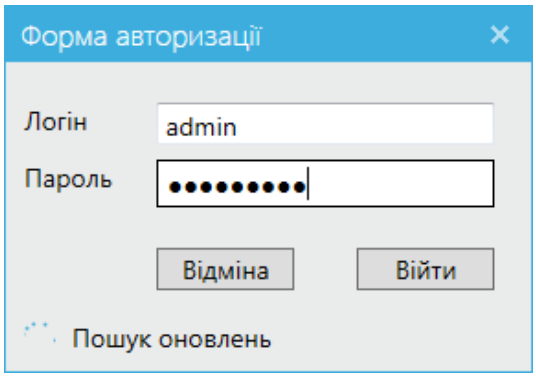


Рисунок 4.12 – Авторизація. Підключення до серверу та пошук оновлень

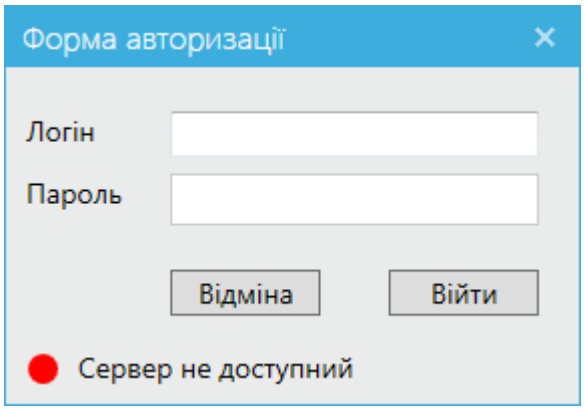


Рисунок 4.13 – Авторизація. Підключення до серверу не було встановлено

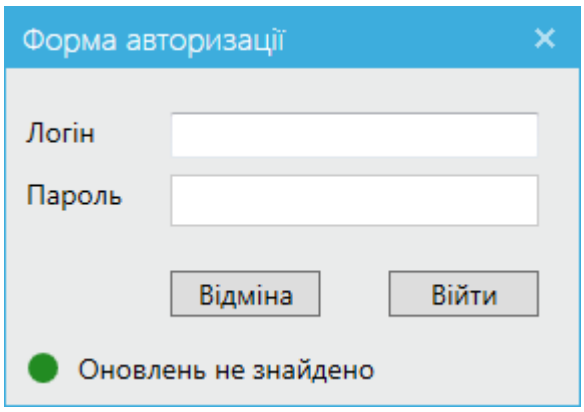


Рисунок 4.14 – Авторизація. Оновлень не знайдено

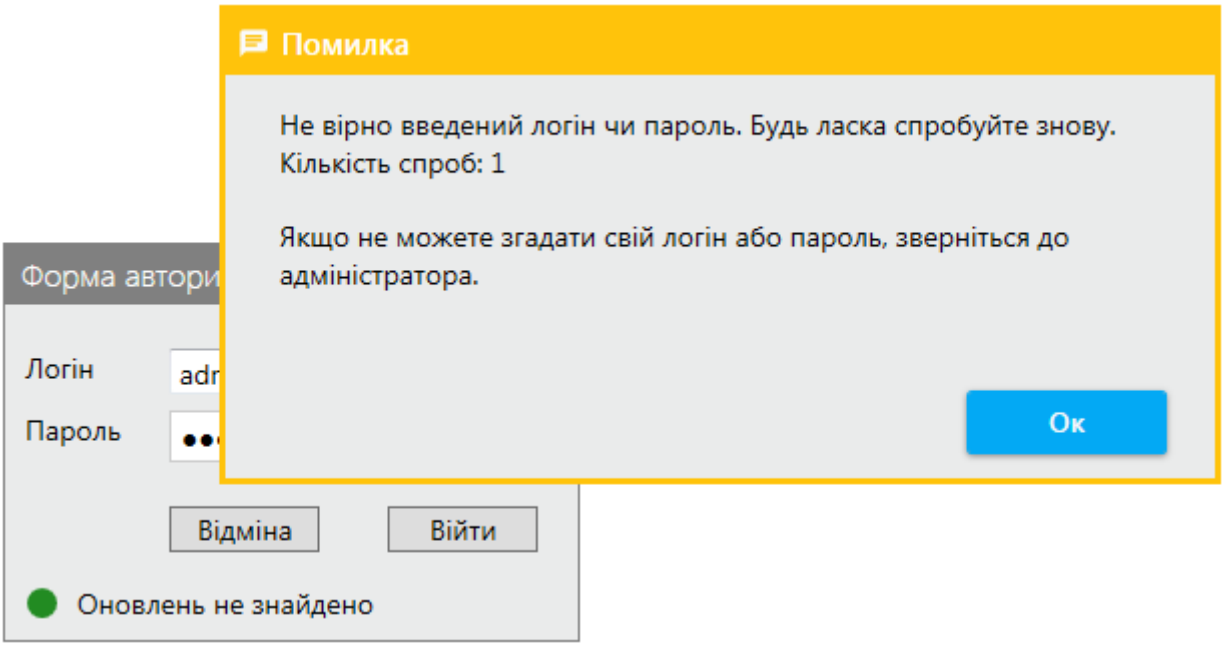


Рисунок 4.15 – Авторизація. Не вірно введений логін або пароль

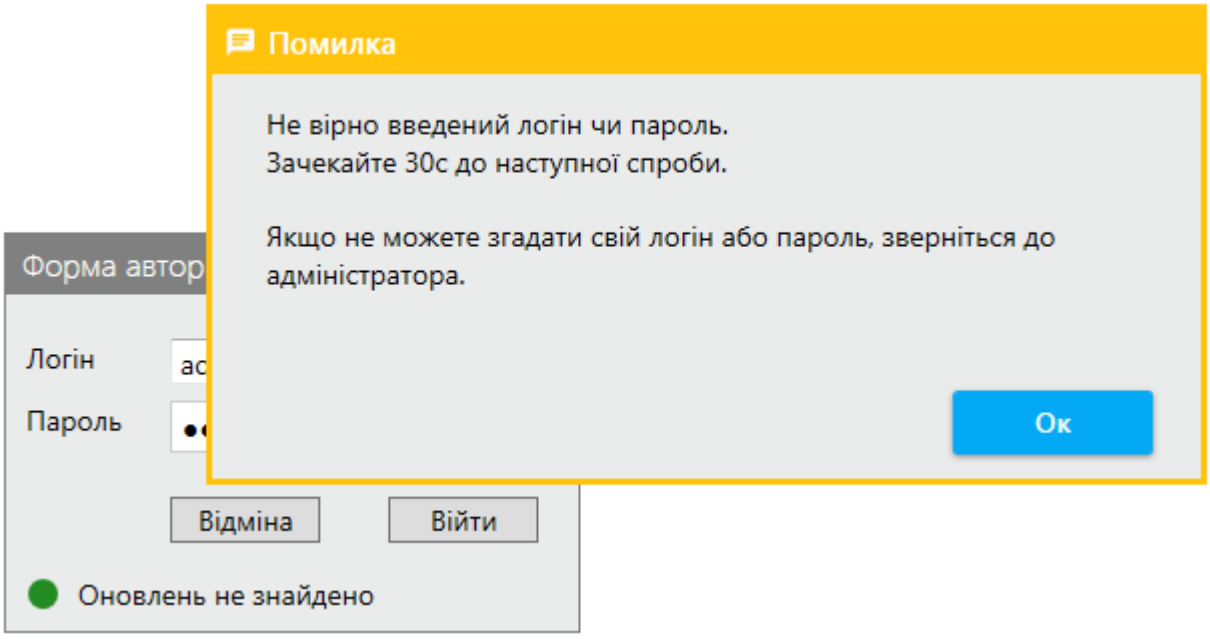


Рисунок 4.16 – Авторизація. Не вірно введений логін або пароль декілька разів

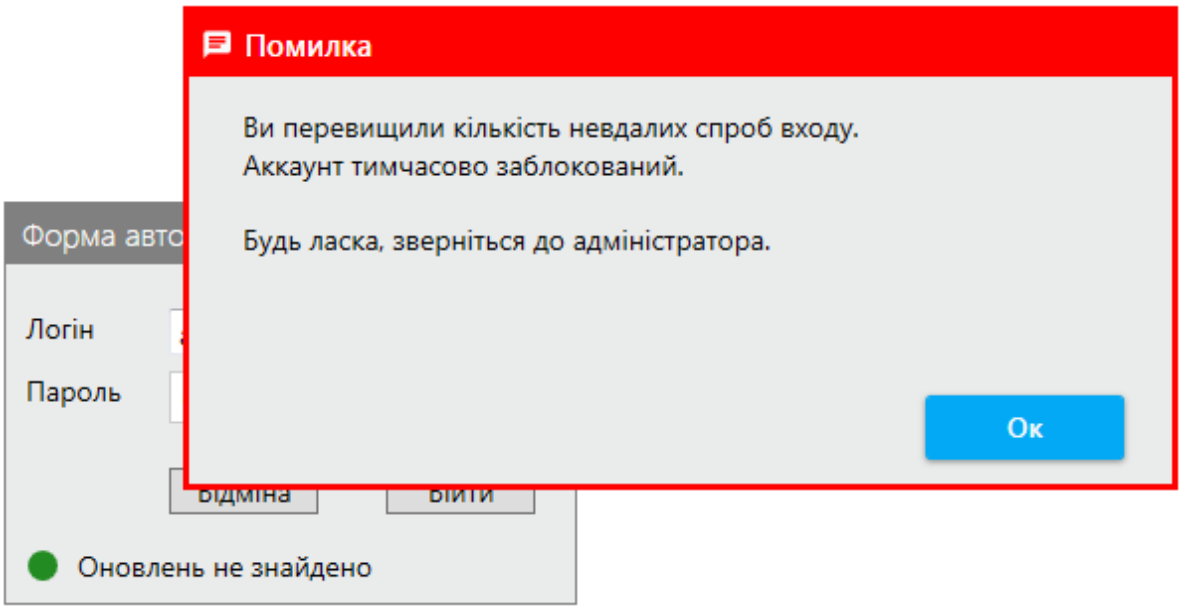


Рисунок 4.17 – Авторизація. Перевищена кількість спроб входу в систему

Під час користування додатку, першу форму, що бачить користувач – форма авторизації (рис. 12). Якщо з'єднання із сервером встановлено, буде спроба знайти оновлення для клієнтського додатку. Якщо оновлення знайдено, воно буде встановлено автоматично. Якщо ні – то в графі "Статус" буде надпис "Оновлень не знайдено" (рис. 14). Якщо ж встановити підключення до серверу додатку неможливо, надпис зміниться на "Сервер не доступний" (рис. 13). При введенні неправильних логіну або паролю 1-3 рази буде відображатись повідомлення про не вірні дані входу (рис. 15), при не правильному вводі 4-7

разів в графі статус буде відображатись кількість часу до наступної спроби (рис. 16) або, при перевищенні кількості спроб, аккаунт користувача буде заблоковано (рис. 17) і розблокувати його зможе лише адміністратор.

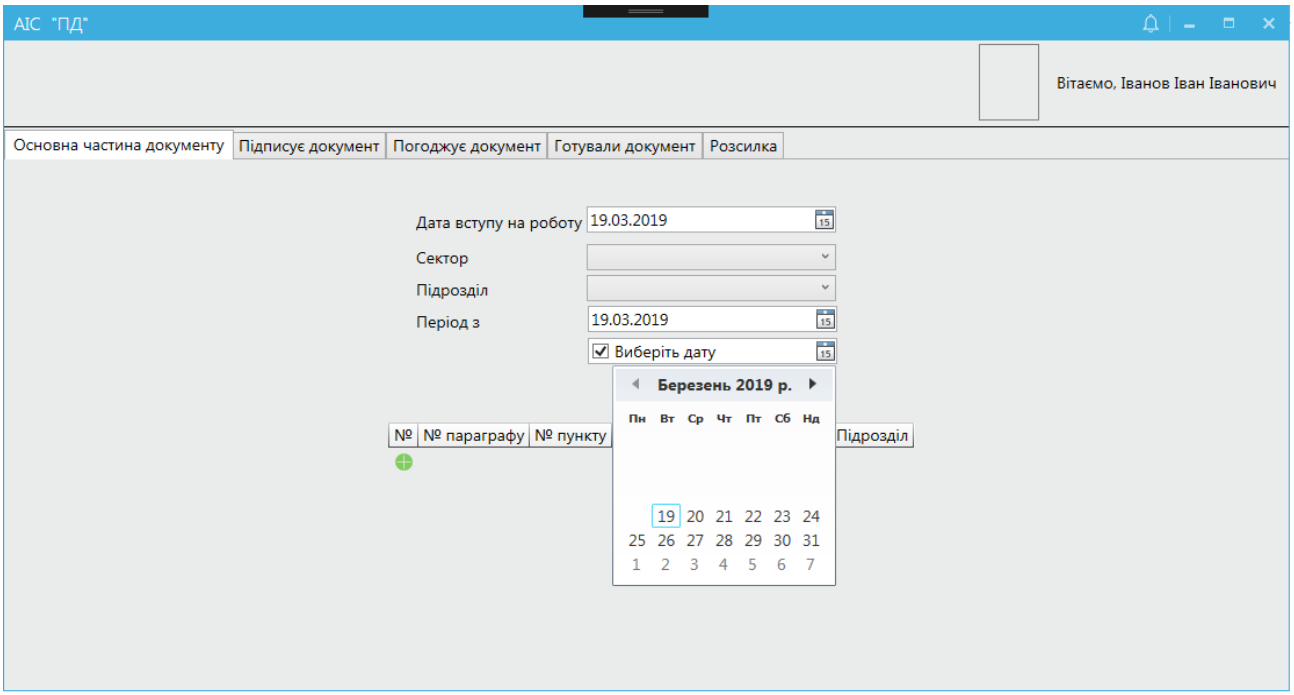


Рисунок 4.18 – Неправильний формат вводу. Основна частина документу. Період роботи

Під час створення документу, деякі поля потребують лише чисельні значення, тож задля зручності користувача, в ці поля можна вводити лише числа, а при натисканні кнопки “Зберегти” всі дані на формі перевіряються. Проте, в деяких місцях (наприклад, рис. 18) можна задавати часовий проміжок. Тож задля зручності користувача, додаток прибирає дати які логічно не можуть бути в полі.

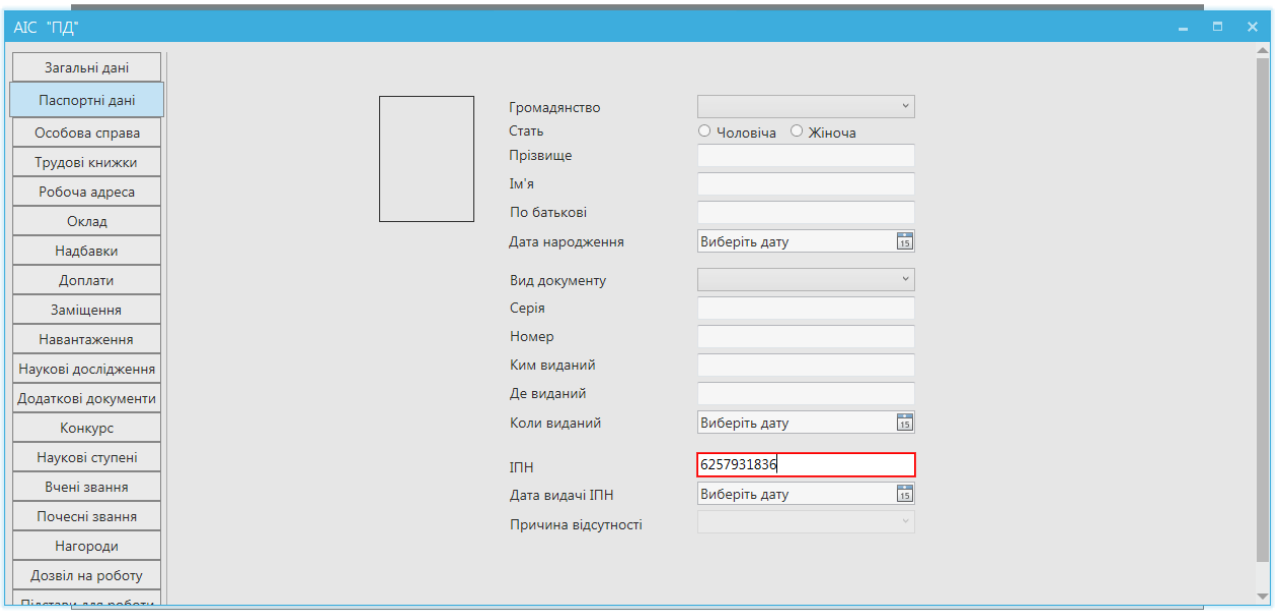


Рисунок 4.19 – Ввід ІПН. Паспортні дані. Не вірно введений ІПН

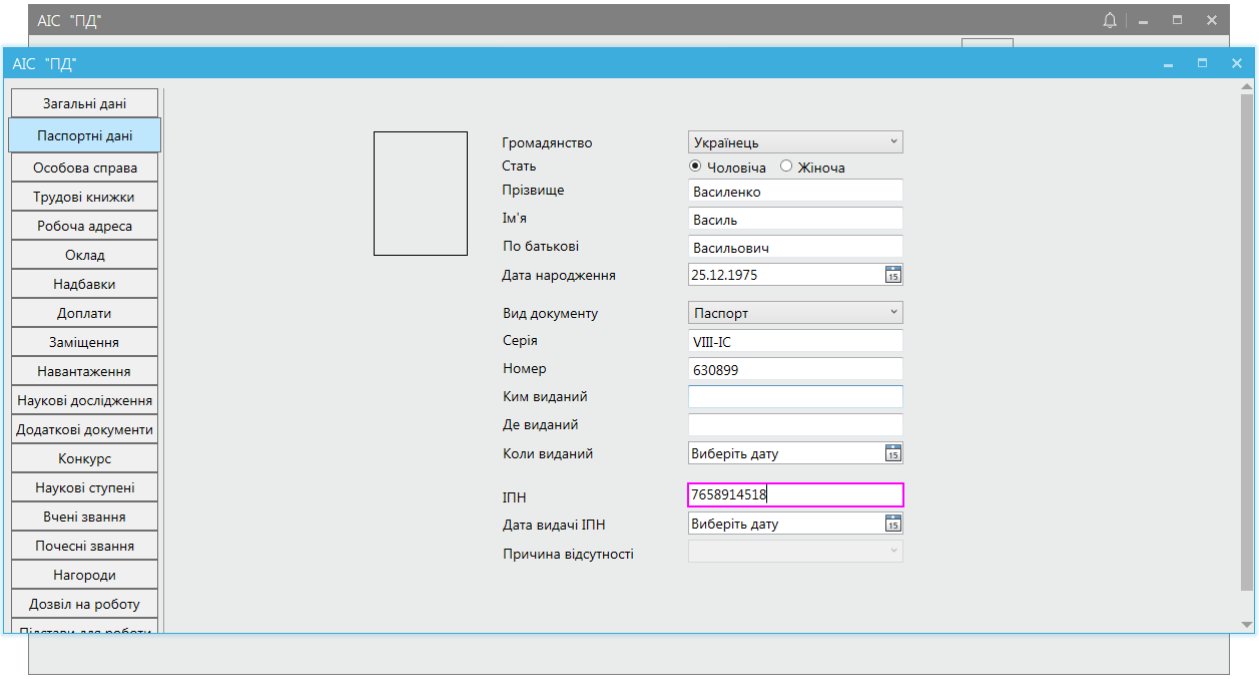


Рисунок 4.20 – Паспортні дані. Можливо, ІПН не належить вказаній особі

Під час вводу паспортних даних, потрібно вказувати індивідуальний податковий номер особи (рис. 19). Якщо при вводі ІПН, він не може існувати взагалі, то починається анімація із затемненням інших контролів та виділенням поля з ІПН для звернення на нього уваги (рис. 19). При неможливості встановлення введеного ІПН вказаній особі, введене поле буде виділятись, проте інші контроли не будуть затемнюватись (рис. 20).

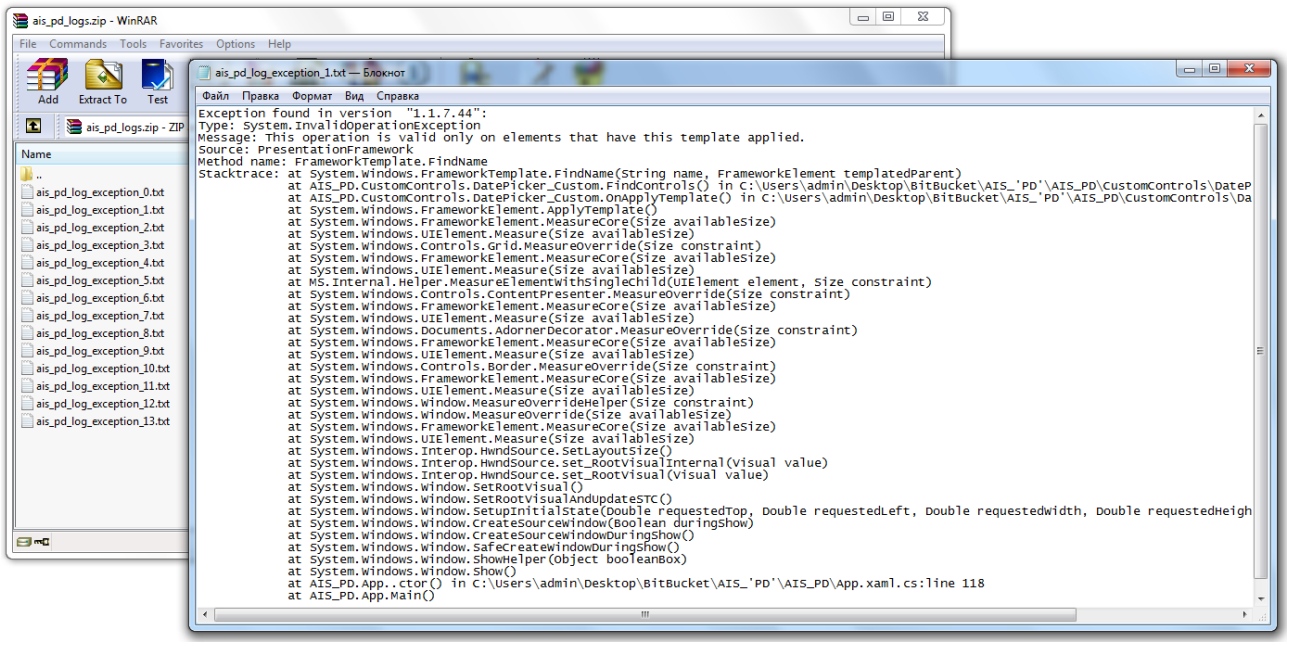


Рисунок 4.21 – Збір критичних помилок (exceptions). Приклад логу помилки в .zip архіві

Під час роботи додатку, можливі ситуації що не були передбачені при його проектуванні та розробці. Під час цих ситуацій можуть виникати критичні помилки (exceptions). Для зручного повідомлення про помилки, додаток автоматично записує критичні помилки у файл, а файл запаковує в .zip архів (рис. 21). Якщо критична помилка не призвела до закриття додатку, архів (з одним файлом) буде переданий до серверу негайно, якщо він доступний. Якщо додаток аварійно завершив свою роботу, то архів буде відправлений за першої нагоди. Доки помилки не передані до серверу, всі вони будуть додаватись до архіву, доки він не буде відправлений до серверу. Тож користувачу не потрібно обтяжувати себе передачою логу адміністраторам та розробникам.

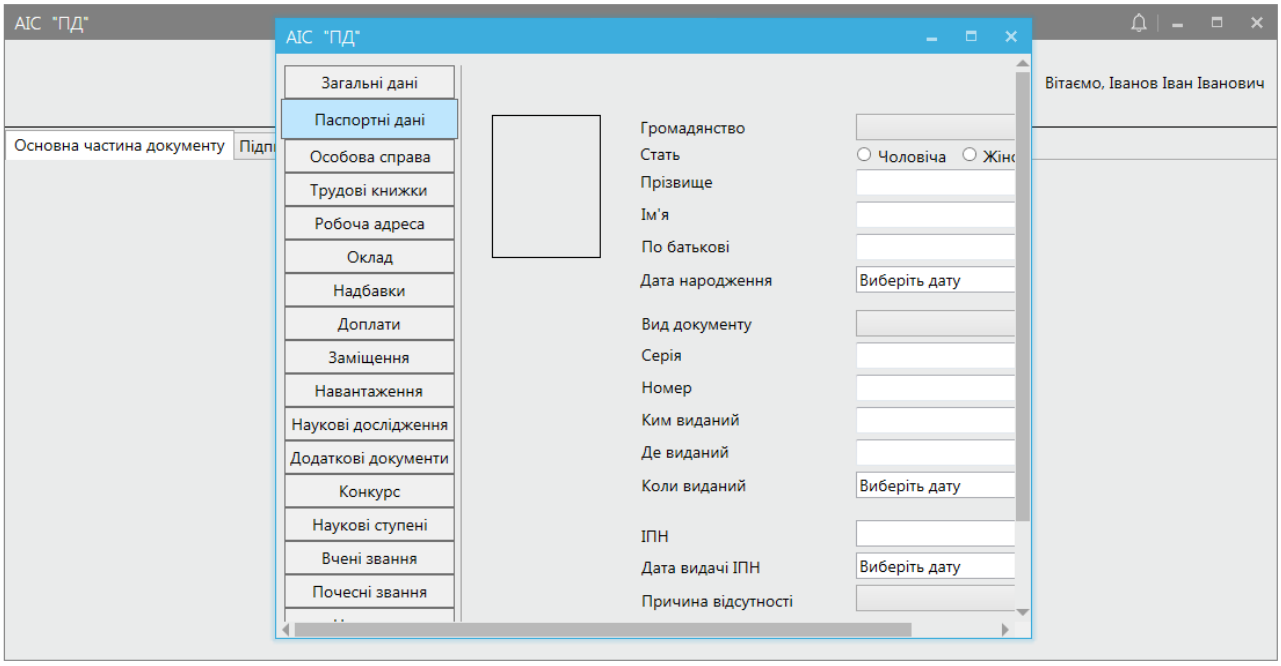


Рисунок 4.22 – Адаптивність інтерфейсу. Приклад адаптивності інтерфейсу

Під час роботи, працівник може працювати в системі з будь-якого комп’ютера, якщо він має інтернет. Це було враховано під час розробки інтерфейсу та було прийняте рішення, що розміри всіх контролів, сторінок та форм будуть однаковими, а головне вікно матиме можливість прокрутки, при зміні його розміру.

## Висновки до розділу

Після проектування, згідно до вимог, автоматизованої інформаційної системи «Проект документу» був розроблений додаток користувача. Приклади інтерфейсу були розглянуті за допомогою прикладу роботи за додатком. Були протестовані основні елементи додатку.

# РОЗДІЛ 5. МАРКЕТИНГОВИЙ АНАЛІЗ СТАРТАП–ПРОЕКТУ

Стартап як форма малого ризикового (венчурного) підприємництва впродовж останнього десятиліття набула широкого розповсюдження у світі через зниження бар'єрів входу в ринок (із появою Інтернету як інструменту комунікацій та збуту стало простіше знаходити споживачів та інвесторів, займатись пошуком ресурсів, перетинати кордони між ринками різних країн), і вважається однією із наріжних складових інноваційної економіки, оскільки за рахунок мобільності, гнучкості та великої кількості стартап-проектів загальна маса інноваційних ідей зростає.

Проте створення та ринкове впровадження стартап-проектів відзначається підвищеною мірою ризику, ринково успішними стає лише невелика частка, що за різними оцінками складає від 10% до 20%. Ідея стартап-проекту, взята окремо, не вартує майже нічого: головним завданням керівника проекту на початковому етапі його існування є перетворення ідеї проекту у працюючу бізнес-модель, що починається із формування концепції товару (послуги) для визначеної клієнтської групи за наявних ринкових умов.

Розроблення та виведення стартап-проекту на ринок передбачає здійснення низки кроків, в межах яких визначають ринкові перспективи проекту, графік та принципи організації виробництва, фінансовий аналіз та аналіз ризиків і заходи з просування пропозиції для інвесторів. Узагальнено етапи розроблення стартап-проекту можна подати таким чином.

## 5.1 Етапи розроблення стартап-проекту

1. Маркетинговий аналіз стартап-проекту

В межах цього етапу:

- розробляється опис самої ідеї проекту та визначаються загальні на- прями використання потенційного товару чи послуги, а також їх відмінність від конкурентів;
- аналізуються ринкові можливості щодо його реалізації;

• на базі аналізу ринкового середовища розробляється стратегія ринкового впровадження потенційного товару в межах проекту.

2. Організація стартап-проекту

В межах цього етапу:

• складається календарний план-графік реалізації стартап-проекту;

• розраховується потреба в основних засобах та нематеріальних активах;

• визначається плановий обсяг виробництва потенційного товару, на основі чого формулюється потреба у матеріальних ресурсах та персоналі;

• розраховуються загальні початкові витрати на запуск проекту та планові загальногосподарські витрати, необхідні для реалізації проекту.

3. Фінансово-економічний аналіз та оцінка ризиків проекту

В межах цього етапу:

• визначається обсяг інвестиційних витрат;

• розраховуються основні фінансово-економічні показники проекту (обсяг виробництва продукції, собівартість виробництва, ціна реалізації, податкове навантаження та чистий прибуток) та визначаються показники інвестиційної привабливості проекту (запас фінансової міцності, рентабельність продажів та інвестицій, період окупності проекту);

• визначається рівень ризикованості проекту, визначаються основні ризики проекту та шляхи їх запобігання (реагування на ризики).

4. Заходи з комерціалізації проекту

Цей етап спрямовано на пошук інвесторів та просування інвестиційної пропозиції (оферти). Він передбачає:

• визначення цільової групи інвесторів та опису їх ділових інтересів;

• складання інвест-пропозиції (оферти): стислої характеристики проекту для попереднього ознайомлення інвестора із проектом;

• планування заходів з просування оферти: визначення комунікаційних каналів та площадок та планування системи заходів з просування в межах обраних каналів;

• планування ресурсів для реалізації заходів з просування оферти.

## 5.2 Опис ідеї проекту

В межах підпункту слід послідовно проаналізувати та подати у вигляді таблиць:

- зміст ідеї (що пропонується);
- можливі напрямки застосування;
- основні вигоди, що може отримати користувач товару (за кожним напрямком застосування);
- чим відрізняється від існуючих аналогів та замінників;

Перші три пункти подаються у вигляді таблиці (табл. 5.1) і дають цілісне уявлення про зміст ідеї та можливі базові потенційні ринки, в межах яких потрібно шукати групи потенційних клієнтів.

Таблиця 5.1. – Опис ідеї стартап–проекту

| Зміст ідеї | Напрямки застосування | Вигоди для користувача |
|---|---|---|
| Підвищення ефективності обробки інформаційних потоків в системі документообігу університету | Корпорації;<br>Університети;<br>Підприємства; | Економічність;<br>Надійність;<br>Безпечність; |

Аналіз потенційних техніко-економічних переваг ідеї (чим відрізняється від існуючих аналогів та замінників) порівняно із пропозиціями конкурентів передбачає:

- визначення переліку техніко-економічних властивостей та характеристик ідеї (орієнтований можливий перелік властивостей та характеристик подано у додатку А);
- визначення попереднього кола конкурентів (проектів-конкурентів) або товарів-замінників чи товарів-аналогів, що вже існують на ринку, та проводиться збір інформації щодо значень техніко-економічних показників для ідеї власного проекту та проектів-конкурентів відповідно до визначеного вище переліку;

• проводиться порівняльний аналіз показників: для власної ідеї визначаються показники, що мають а) гірші значення (W, слабкі); б) аналогічні (N, нейтральні) значення; в) кращі значення (S, сильні) (табл. 2).

Таблиця 5.2 – Визначення сильних, слабких та нейтральних характеристик ідеї проекту

| № | Техніко–економічні характеритики ідеї | Продукція конкурентів | | | W (слабка сторона) | N (нейтральна сторона) | S (сильна сторона) |
|---|---|---|---|---|---|---|---|
| | | Мій проект | *1С* | *FossDoc* | | | |
| 1 | Собівартість | Низька | Висока | Середня | | | + |
| 2 | Зручність використання | Висока | Вище за середню | Низька | | | + |
| 3 | Інформаційна безпека | Висока, модифіковані способи електронного підпису | Середня | Середня | | + | |

Продовження таблиці 5.2

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 4 | Масштабованість системи | Система може бути швидко переписана під потреби користувачів | Систему можливо перенала штувати під потреби користувачів | Систему не можливо перенала штувати | | | + |
| 5 | Інтеграція з системами університету | Система легко інтегрується з існуючими | Систему можна інтегрувати за допомогою зовнішніх скриптів | Система не інтегрується | | | + |

Визначений перелік слабких, сильних та нейтральних характеристик та властивостей ідеї потенційного товару є підґрунтям для формування його конкурентоспроможності

## 5.3 Технологічний аудит ідеї проекту

В межах даного підрозділу необхідно провести аудит технології, за допомогою якої можна реалізувати ідею проекту (технології створення товару). Визначення технологічної здійсненності ідеї проекту передбачає аналіз таких складових (табл. 5.3):

- за якою технологією буде виготовлено товар згідно ідеї проекту?
- чи існують такі технології, чи їх потрібно розробити/доробити?
- чи доступні такі технології авторам проекту?

Таблиця 5.3 – Технологічна здійсненність ідеї проекту

| № | Ідея проекту | Технології її реалізації | Наявність технологій | Доступність технологій |
|---|---|---|---|---|
| 1 | Електронно-цифровий підпис | Ed22519 | + | + |
| | | Argon2 | + | + |
| 2 | Автоматизований “потік” документу | .Net 4.8 | + | + |

## 5.4 Аналіз ринкових можливостей запуску стартап–проекту

Спочатку було проведено аналіз попиту: наявність попиту, обсяг, динаміка розвитку ринку (таблиця 5.4)

Таблиця 5.4 – Попередня характеристика потенційного ринку

| № | Показники стану ринку | Характеристика |
|---|---|---|
| 1 | Кількість головних гравців, од | 200+ |
| 2 | Загальний обсяг продаж, грн./ум.од | 261 830 грн. на 1 продукт |
| 3 | Динаміка ринку | Розвивається |
| 4 | Наявність обмежень для входу | Немає |
| 5 | Специфічні вимоги до стандартизації та сертифікації | Немає |
| 6 | Середня норма рентабельності в галузі або по ринку, % | 40 % |

Висновок: враховуючи кількість головних гравців по ринку, зростаючу динаміку ринку, невелику кількість конкурентів та середню норму рентабельності можна зробити висновок, що на даний момент, ринок для входження стартап–продукту є привабливим.

Далі були визначені потенційні групи клієнтів, їх характеристики, та сформовано орієнтовний перелік вимог до товару для кожної групи (таблиця 5.5).

Таблиця 5.5 – Характеристика потенційних клієнтів стартап–проекту

| № | Потреба, що формує ринок | Цільова аудиторія | Відмінності у поведінці цільових груп клієнтів | Вимоги споживачів до товару |
|---|---|---|---|---|
| 1 | Зменшення паперового документообігу | Корпорації; Університети; Підприємства; | В кожної цільової аудиторії свої бізнес-процеси документу | Надійність даних; Швидкість роботи в системі |
| 2 | Надійність та швидкість електронного документообігу | | | |

Після визначення потенційних груп клієнтів було проведено аналіз ринкового середовища: складено таблиці факторів, що сприяють ринковому впровадженню проекту, та факторів, що йому перешкоджають (таблиця 5.6, 5.7).

Таблиця 5.6 – Фактори загроз

| № | Фактор | Зміст загрози | Можлива реакція компанії |
|---|---|---|---|
| 1 | Конкуренція | Поява більш гнучкого продукту | 1. Передбачити додаткові переваги власного проекту для того, щоб повідомити про них саме після виходу |

| | | | |
|---|---|---|---|
| | | | міжнародної компанії на ринок. |

Продовження таблиці 5.6

| | | | |
|---|---|---|---|
| | | | 2. Обрати нову цільову аудиторію і зосередитися на ній |
| 2 | Безпека | Поширення квантових комп’ютерів | Немає |

Таблиця 5.7 – Фактори можливостей

| № | Фактор | Зміст можливості | Можлива реакція компанії |
|---|---|---|---|
| 1 | Новий продукт | Вихід на ринок, Надання нових рішень у сфері | Розробка нової функціональності; Вихід нової продукції на ринок; Надання різноманітних типів ліцензій в залежності від потреб користувача \ замовника. |
| 2 | Вихід аналогу | Надати продукт з певними характеристиками та можливостями що відсутні у компаній конкурентів | Аналіз ринку та користувачів задля задоволення їх потреб та надання функціональності у найкоротші строки за ціну, котра є дешевшою ніж у продуктів–замінників. |
| 3 | Зворотній зв’язок від користувачів | Можливість отримання необхідної інформації для вдосконалення продукту | Наявність вхідних даних та реакція на них з боку команди розробників задля задоволення потреб та бажань кінцевих користувачів системи кешування даних. |
| 4 | Грошова винагорода за рекламу | При достатньому попиту на систему кешування даних можлива комерціалізація продукту на основі реклами задля отримання грошової винагороди для подальшого розвиту продукту та оплати заробітної плати | Точкова комерціалізація продукту; Введення реклами; Ведення додаткових коштів у проект задля його подальшого розвитку. |

| | | | |
|---|---|---|---|
| | | працівникам | |

Далі було проведено аналіз пропозиції: визначили загальні риси конкуренції на ринку (таблиця 5.8)

Таблиця 5.8 – Ступеневий аналіз конкуренції на ринку

| № | Особливості конкурентного середовища | В чому проявляється дана характеристика | Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною) |
|---|---|---|---|
| 1 | Тип конкуренції: монополістична | Товар від кожної компанії на ринку, являється недосконалим замінником товару, реалізованого іншими фірмами;<br>На ринку є умови для входу та виходу;<br>Ціна корелює між суперниками; | Розробка продукту з характеристиками, які покривають сфери вживання що не покривають інші товари–замінники;<br>Кореляція цін у відповідності до товарів замінників;<br>Різни типи ліцензій. |
| 2 | Рівень конкурентної боротьби: світовий | Всі продукти замінники розроблялись інтернаціональними командами з різних куточків світу, продукти не належать до певної держави, а належать команді розробників | Вихід на ринок збуту продукту з клієнто–необхідною функціональністю;<br>Налагодження маркетингу на основних Інтернет ресурсах задля охоплення великої кількості потенційних користувачів;<br>Надання бета–версій продукту. |
| 3 | Галузева ознака: універсальна | Даний тип продукту може використовуватися для будь-яких організацій | Надання зручного, інтуїтивно зрозумілого інтерфейсу;<br>Підтримка всім відомих методів взаємодії з середовищем розробки;<br>Наявність документації та он–лайн |

| | | | |
|---|---|---|---|
| | | | підтримки. |

Продовження таблиці 5.8

| | | | |
|---|---|---|---|
| 4 | Конкуренція за видами товарів: товарно–видова | Дана конкуренція – конкуренція між товарами одного виду. | Впровадження функціональності яка відсутня у товарів–замінників; Спрощення інтерфейсів; Надання підтримки. |
| 5 | Характер конкурентних переваг: цінова та не цінова | Цінові переваги – точкова комерціалізація; Не цінова – надання функціональності, що відсутня у товарах–замінниках. | Надання платних ліцензій лише на критично важливу функціональність для клієнта з певним строком підтримки, що зазначена у відповідній ліцензії; Впровадження унікальної функціональності. |
| 6 | За інтенсивністю: марочна | Наявність унікального знаку що відрізняє даний продукт від продуктів–замінників | Впровадження власної назви та власного знаку. |

Було проведено аналіз конкуренції у галузі за моделлю М. Портера (таблиця 5.9).

Таблиця 5.9 – Аналіз конкуренції в галузі за М. Портером

| | Прямі конкуренти в галузі | Потенційні конкуренти | Постачальники | Клієнти | Товари–замінники |
|---|---|---|---|---|---|
| Складові аналізу | 1С, MeDoc, FossDoc | Наявність вже існуючих рішенньь | - | Співробітники організацій | - |
| Висновки | Інтенсивна конкурентна боротьба | Є можливості виходу на ринок, але є і конкуренти. Строки – 1 рік. | - | Клієнти диктують усі умови роботи на ринку. | - |

Проаналізувавши можливості роботи на ринку з огляду на конкурентну ситуацію можна зробити висновок: оскільки кожний з існуючих продуктів не впливає у великій мірі на поточну ситуацію на ринку в цілому, кожний з існуючих продуктів має свою специфічну сферу використання та свої позитивні та негативні сторони щодо рішення певних типів задач, то робота та вихід на даний ринок є можливою і реалізованою задачею.

Для виходу на ринок продукт повинен мати функціонал що відсутній у продуктів–аналогів, повинен задовольняти потреби користувачів, мати необхідний та достатній функціонал з конфігурування, підтримку зі сторони розробників та можливість розробки спеціального функціоналу за відповідною ліцензією.

Таблиця 5.10 – Обґрунтування факторів конкурентоспроможності

| № | Фактор конкурентоспроможності | Обґрунтування |
|---|---|---|
| 1 | Економічність | Дана система має гнучке ціноутворення |
| 2 | Надійність | На ринку немає такого продукту |
| 3 | Безпечність | Використовується стандарти захисту та модифікований алгоритм ЕЦП |

За визначеними факторами конкурентоспроможності (таблиця 5.10) проведено аналіз сильних та слабких сторін стартап-проекту (таблиця 5.11).

Таблиця 5.11 – Порівняльний аналіз сильних та слабких сторін

| № | Фактор конкурентоспроможності | Бали 1–20 | Рейтинг товарів–конкурентів у порівнянні з запропонованим | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | –3 | –2 | –1 | 0 | +1 | +2 | +3 |
| 1 | Економічність | 14 | | | | | | | + |
| 2 | Надійність | 19 | | | | | + | | |
| 3 | Безпечність | 17 | | | | | | + | |

Фінальним етапом ринкового аналізу можливостей впровадження проекту є складання SWOT-аналізу (матриці аналізу сильних (Strength) та

слабких (Weak) сторін, загроз (Troubles) та можливостей (Opportunities) (таблиця 5.12) на основі виділених ринкових загроз та можливостей, та сильних і слабких сторін (таблиця 5.11).

Таблиця 5.12 – SWOT аналіз стартап–проекту

| Сильні сторони (S): | Слабкі сторони (W): |
|---|---|
| Контроль за здійсненням витрат, пошук можливостей щодо їхнього зниження;<br>Інвестиційна привабливість підприємства;<br>Зважена цінова політика;<br>Врахування потреб споживачів. | Частка ринку;<br>Організація системи комунікацій. |
| Можливості (O): | Загрози (T): |
| Зростання грошових доходів;<br>Застосування сучасних технологій | Недосконаліть та змінюваність законодавства;<br>Інфляційні процеси;<br>Високий рівень безробіття. |

На основі SWOT-аналізу було розроблено альтернативи ринкової поведінки (перелік заходів) для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок (див. таблицю 5.9, аналіз потенційних конкурентів). Визначені альтернативи були проаналізовані з точки зору строків та ймовірності отримання ресурсів (таблиця 5.13).

Таблиця 5.13 – Альтернативи ринкового впровадження стартап–проекту

| № | Альтернатива (орієнтовний комплекс заходів) ринкової поведінки | Ймовірність отримання ресурсів | Строки реалізації |
|---|---|---|---|
| 1 | Розробка програмного продукту, PR, просування бренду | вище середньої | 1 рік |
| 2 | Стратегія підсилення сильних сторін за рахунок ринкових можливостей. | висока | 6 місяців |
| 3 | Стратегія компенсації слабких сторін наявними ринковими | середня | 2 роки |

| | | | |
|---|---|---|---|
| | можливостями. | | |

## 5.5 Розроблення ринкової стратегії проекту

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку: було проведено опис цільових груп потенційних споживачів (таблиця 5.14).

Таблиця 5.14 – Вибір цільових груп потенційних споживачів

| № | Опис профілю цільової групи потенційних клієнтів | Готовність споживачів сприйняти продукт | Орієнтовний попит в межах цільової групи (сегменту) | Інтенсивність конкуренції в сегменті | Простота входу у сегмент |
|---|---|---|---|---|---|
| 1 | Організації | Висока | Специфічна | Вище середньої | Високий бар’єр входу у галузь |

Відповідно до проведеного аналізу можна зробити висновок, що підходящою цільовою групою для розповсюдження даного програмного продукту є будь-які організації. Відповідно до стратегії охоплення ринку збуту товару обрано стратегію масового маркетингу, оскільки надається стандартизований продукт з можливістю розширення функціональності за домовленістю (відповідно до ліцензії).

Таблиця 5.15 – Визначення базової стратегії розвитку

| Обрана альтернатива розвитку проекту | Стратегія охоплення ринку | Ключові конкурентоспроможні позиції відповідно до обраної альтернативи |
|---|---|---|
| Стратегія диференціації | Передбачає надання товару важливих з точки зору споживача відмітних властивостей, які роблять товар відмінним від товарів конкурентів. Така відмінність може базуватися на об’єктивних або суб’єктивних, | Реалізація цієї стратегії вимагає, як правило, більш високих витрат. Проте успішна диференціація дозволяє компанії домогтись більшої рентабельності за рахунок того, що ринок готовий прийняти |

| | | |
|---|---|---|
| | відчутних і невідчутних властивостях товару | більш високу ціну |

Продовження таблиці 5.15

| | | |
|---|---|---|
| | (у ширшому розумінні – комплексі маркетингу), бути реальною або уявною. | (цінову премію бренду). |

На основі вимог споживачів з обраних сегментів до постачальника (стартап-компанії) та до продукту (див. таблицю 5.5), а також в залежності від обраної базової стратегії розвитку (таблиця 5.15) та стратегії конкурентної поведінки (таблиця 5.16) розроблено стратегію позиціонування (таблиця 5.17), що полягає у формуванні ринкової позиції (комплексу асоціацій), за яким споживачі мають ідентифікувати торгівельну марку/проект.

Таблиця 5.16. – Визначення базової стратегії конкурентної поведінки

| Чи є проект «першопрохідцем» на ринку | Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів? | Чи буде компанія копіювати основні характеристики товару конкурента, які? | Стратегія конкурентної поведінки |
|---|---|---|---|
| Ні | Так | Частково | Стратегія заняття конкурентної ніші |

Таблиця 5.17 – Визначення стратегії позиціонування

| № | Вимоги до товару цільової аудиторії | Базова стратегія розвитку | Ключові конкурентоспроможні позиції власного стартап–проекту | Вибір асоціацій, які мають сформувати комплексну позицію власного проекту |
|---|---|---|---|---|
| 1 | Відповідність чинним нормативам | Заняття конкурентної ніші | Реалізація цієї стратегії вимагає, як правило, більш високих витрат. | Унікальність<br>Доступна ціна<br>Реалізація нових |

| | | | | |
|---|---|---|---|---|
| | | | | методів |

Продовження таблиці 5.17

| | | | | |
|---|---|---|---|---|
| | | | Проте успішна диференціація дозволяє компанії домогтись більшої рентабельності за рахунок того, що ринок готовий прийняти більш високу ціну (цінову премію бренду). | |

Відповідно до проведеного аналізу можна зробити висновок, що стартап–компанія вибирає як базову стратегію розвитку – стратегію диференціації, як базову стратегію конкурентної поведінки – стратегію заняття конкурентної ніші.

## 5.6 Розроблення маркетингової програми стартап–проекту

Сформовано маркетингову концепцію товару, який отримає споживач. Для цього у таблиці 5.18 підсумовано результати попереднього аналізу конкурентоспроможності товару.

Таблиця 5.18 – Визначення ключових переваг концепції потенційного товару

| № | Потреба | Вигода, яку пропонує товар | Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити |
|---|---|---|---|
| 1 | Ціна | Низька ціна | Ціна значно нижче за аналоги |
| 2 | Надійність | Висока надійність | Слідування європейським стандартам безпеки інформації |

Розроблено трирівневу маркетингову модель товару: уточнюється ідея продукту та/або послуги, його фізичні складові, особливості процесу його надання (таблиця 5.19).

Таблиця 5.19 – Опис трьох рівнів моделі товару

| Рівні товару | Сутність та складові |
|---|---|

| | | | |
|---|---|---|---|
| I. Товар за задумом | Програмний продукт дозволяє автоматизувати документообіг в організації | | |
| II. Товар у реальному виконанні | Властивості/характеристики | М/Нм | Вр/Тх /Тл/Е/Ор |
| | 1. Надійність<br>2. Простота у використанні<br>3. Низька ціна | - | - |
| | Якість: слідування європейським стандартам | | |
| | Пакування немає | | |
| | Марка: КБІС – АІС “ПД” | | |

Продовження таблиці 5.19

| | |
|---|---|
| III. Товар із підкріпленням | До продажу: Пробна безкоштовна версія |
| | Після продажу: Розширення функціоналу |
| За рахунок чого потенційний товар буде захищено від копіювання: Ліцензія | |

Наступним кроком є визначення цінових меж, якими необхідно керуватись при встановленні ціни на потенційний товар (остаточне визначення ціни відбувається під час фінансово-економічного аналізу проекту), яке передбачає аналіз ціни на товари-аналоги або товари субститути, а також аналіз рівня доходів цільової групи споживачів (таблиця 5.20). Аналіз проведено експертним методом.

Таблиця 5.20 – Визначення меж встановлення ціни

| № п/п | Рівень цін на товари-замінники | Рівень цін на товари-аналоги | Рівень доходів цільо- вої групи споживачів | Верхня та нижня межі вста-новлення ціни на то-вар/послугу |
|---|---|---|---|---|
| 1 | 350000-500000 грн | 350000-500000 грн | 200 млн грн\рік | 250000-290000 грн |

Наступним кроком є визначення оптимальної системи збуту, в межах якого приймається рішення (табл. 5.21):

- проводити збут власними силами або залучати сторонніх посередників (власна або залучена система збуту);
- вибір та обґрунтування оптимальної глибини каналу збуту;
- вибір та обґрунтування виду посередників.

Таблиця 5.21 – Формування системи збуту

| № п/п | Специфіка закупівельної поведінки цільових кліє- нтів | Функції збуту, які має виконувати постачаль-ник товару | Глибина каналу збуту | Оптимальна система збуту |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| 1 | Мінімальна кількість посередників | Організовувати широку мережу збуту товару | 3 | непряма |

Останньою складової маркетингової програми є розроблення концепції маркетингових комунікацій, що спирається на попередньо обрану основу для позиціонування, визначену специфіку поведінки клієнтів (таблиця 5.22).

Таблиця 5.22 – Концепція маркетингових комунікацій

| № п/п | Специфіка поведінки цільових клієнтів | Канали комунікацій, якими користуються цільові клієнти | Ключові позиції, обрані для позиціонування | Завдання рекламного повідомлення |
|---|---|---|---|---|
| 1 | Обережний вибір потенційних контрагентів, що зумовлено особливістю ринку | Інтернет-розсилки | Технологія | Привернути увагу |

Результатом підрозділу стала ринкова (маркетингова) програма, що включає в себе концепції товару, збуту, просування та попередній аналіз можливостей ціноутворення, спирається на цінності та потреби потенційних клієнтів, конкурентні переваги ідеї, стан та динаміку ринкового середовища, в межах якого впроваджено проект, та відповідну обрану альтернативу ринкової поведінки.

## Висновки до розділу

В даному розділі було проведено аналіз програмного продукту у якості стартап проекту. Можна зазначити що у проекту є можливість комерціалізації, оскільки ринок хоч і великий, але він постійно розвивається, створюються нові додатки як із вузько спеціалізованим, так і широким функціоналом. На ринку наявна конкуренція тому вихід на нього не буде легким. Через те, що стартап проект є повністю програмним, його розробка не потребує витрат на

різноманітні матеріали та обладнання. Оскільки проект має свою цільову аудиторію можна сказати, що подальший його розвиток є доцільним.

# ВИСНОВКИ

1. Проведено аналіз:
   - елементів робочого процесу системи електронного документообігу;
   - європейської практики використання електронних підписів і пов'язаних з ними послуг по сертифікації;
   - основних особливостей електронного підпису;
   - використання електронного цифрового підпису, як засобу захисту інформації;
2. Проаналізовані методи шифрування та зроблений висновок, щодо використання, для досягнення поставленої мети, функцій Argon2.
3. За результатами аналізу процесів документообігу університету спроектовано:
   - контекстну діаграму «Оформлення працівника на роботу»;
   - функціональну декомпозицію системи;
   - діаграму інформаційних потоків.
4. Сформовані вимоги до системи документообігу університету.
5. Розглянуто та обрано основні технології для розробки системи:
   - для розробки БД було обрано MsSQL Server;
   - для розробки клієнтського додатку та серверу – платформу .Net та мову програмування C#.
6. Спроектована структура БД та розроблені спеціалізовані контроли для потреб додатку.
7. Визначені мінімальні вимоги для обладнання користувача.
8. Розроблений інтерфейс користувача.
9. Розроблений додаток користувача.
10. Проведено тестування основних елементів додатку.
11. Проведено аналіз програмного продукту у якості стартап проекту.

За результатами роботи були зроблені висновки, щодо подальший доцільності розвиту проекту.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. Документне забезпечення управлінської діяльності організацій – [Електронный ресурс]. Режим доступу: http://library.dk.rv.ua/depository/methodics/DZUDO.pdf. Дата звернення – 10.05.2019.
2. Martin Fowler. Inversion of Control Containers and the Dependency Injection pattern – [Електронный ресурс]. Режим доступу: https://www.martinfowler.com/articles/injection.html. Дата звернення – 10.05.2019.
3. FAT File Systems – [Електронный ресурс]. Режим доступу: http://www.ntfs.com/fat-systems.htm. Дата звернення – 10.05.2019.
4. John Gossman. Model/View/ViewModel pattern for building WPF apps – [Електронный ресурс]. Режим доступу: https://blogs.msdn.microsoft.com/johngossman/2005/10/08/introduction-to-modelviewviewmodel-pattern-for-building-wpf-apps/. Дата звернення – 10.05.2019.
5. Human Resource Management – [Електронный ресурс]. Режим доступу: https://kissflow.com/hr-process/human-resource-management-guide/. Дата звернення – 10.05.2019.
6. Document Workflow for Review and Approval– [Електронный ресурс]. Режим доступу: https://www.filehold.com/features/optional/workflow. Дата звернення – 10.05.2019.
7. Hardware & Software Survey: April 2019 – [Електронный ресурс]. Режим доступу: https://store.steampowered.com/hwsurvey/. Дата звернення – 10.05.2019.
8. High-speed high-security signatures – [Електронный ресурс]. Режим доступу: https://ed25519.cr.yp.to/ed25519-20110705.pdf. Дата звернення – 10.05.2019.
9. Argon2: the memory-hard function – [Електронный ресурс]. Режим доступу: https://github.com/P-H-C/phc-winner-argon2/blob/master/argon2-specs.pdf. Дата звернення – 10.05.2019.

10. The Password Hashing Competition 2013 – [Електронный ресурс]. Режим доступу: https://eprint.iacr.org/2016/104.pdf. Дата звернення – 10.05.2019.
11. Fast and Tradeoff-Resilient Memory-Hard Functions – [Електронный ресурс]. Режим доступу: https://eprint.iacr.org/2015/430.pdf. Дата звернення – 10.05.2019.
12. Breaking Ed25519 – [Електронный ресурс]. Режим доступу: https://eprint.iacr.org/2015/430.pdf. Дата звернення – 10.05.2019.
13. Ed25519: high-speed high-security signatures – [Електронный ресурс]. Режим доступу: https://ed25519.cr.yp.to/. Дата звернення – 10.05.2019.
14. Public and private key signatures – [Електронный ресурс]. Режим доступу: https://libsodium.gitbook.io/doc/public-key_cryptography/public-key_signatures/. Дата звернення – 10.05.2019.
15. Y. Kornaga, V. Garmatin, A. Hryshko, A. Maksimyuk, V. Gasanov. Protection of an electronic document using a consolidated approach to the application of electronic digital signature // Міжвідомчий науково-технічний збірник «Адаптивні системи автоматичного управління» № 1 (33), 2019 – 3-7 с.
16. Electronic signatures and trust services – [Електронный ресурс]. Режим доступу:https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/545098/beis-16-15-electronic-signatures-guidance.pdf Дата звернення – 10.05.2019.
17. Кодекс законів про працю України від 10.12.71 № 322-VIII – [Електронний ресурс]. Режим доступу : http://zakon2.rada.gov.ua/laws/show/322-08. Дата звернення – 10.05.2019.
18. Directive 1999/93/EC of the European Parliament and of the Council – [Електронний ресурс]. Режим доступу: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:31999L0093. Дата звернення – 10.05.2019.
19. Anna Monus. Web services. SOAP vs REST – [Електронний ресурс]. Режим доступу: https://raygun.com/blog/soap-vs-rest-vs-json/. Дата звернення – 20.04.2019.

20. Yoshitaka Shiotsu. General-purpose language. C# and Java. – [Електронний ресурс]. Режим доступу: https://itvdn.com/en/blog/article/csharp_vs_java. Дата звернення – 19.04.2019.
21. Database engines – [Електронний ресурс]. Режим доступу: https://db-engines.com/en/system/Microsoft+SQL+Server%3BMySQL%3BOracle. Дата звернення – 17.04.2019.
22. Електронний цифровий підпис – [Електронний ресурс]. Режим доступу: https://cryptoworld.su/ecp-elektronnaya-cifrovaya-podpis-polnyj-manual/. Дата звернення – 17.04.2019.
23. Garmatin V.D., Maksymiuk A.V., Tsymbalenko Y.Y. – [Електронний ресурс]. Режим доступу: https://sworld.education/konferua6/73.pdf. Дата звернення – 01.05.2019.
24. Dependency injection in C# – [Електронний ресурс]. Режим доступу: https://itvdn.com/en/blog/article/injection. Дата звернення – 01.04.2019.
25. Juan Francisco Morales Larios. DataAnnotations In Depth – [Електронний ресурс]. Режим доступу: https://www.c-sharpcorner.com/article/dataannotations-in-depth/. Дата звернення – 27.04.2019.

# ДОДАТКИ

ДОДАТОК А

ДОДАТОК Б

ДОДАТОК В

ДОДАТОК Г

ДОДАТОК Д

ДОДАТОК Е

ДОДАТОК Є